VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

VZDÁLENÁ SPRÁVA VÝTAHU

REMOTE LIFT MANAGEMENT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. FILIP WEISSER**
AUTHOR

VEDOUCÍ PRÁCE — **Mgr. ROMAN TRCHALÍK, Ph.D.**
SUPERVISOR

BRNO 2018

Abstrakt

Cílem diplomové práce bylo vytvořit vhodný a snadno použitelný systém, který může pomoct pracovníkům firem při obsluze výtahu. Pro tyto účely bylo nutné nastudovat komunikační rozhraní a protokol řídicí jednotky zapůjčené firmou Výťahy ZEVA s. r. o. Dále bylo nutné najít vhodné zařízení pro zajištění komunikace mezi řídicí jednotkou a serverem. Server slouží pro ukládání a vizualizaci aktuálních dat výtahového systému. Server umožňuje vzdáleně nastavovat parametry výtahů, vyžádat si jejich aktuální konfiguraci a sledovat historická data. Uživatelé přistupují k serveru po autentizaci s různým stupněm oprávnění. Pracovník s požadovaným oprávněním si může v systému nastavit notifikaci, která ho upozorní v případě vzniklých problémů. Tímto způsobem je zkrácena reakční doba pracovníků, což má za následek rychlejší odstranění poruch.

Abstract

The goal of this master thesis was create suitable and easy to use software, which can help workers with maintaining of elevator. For this purpose it was necessary to study communication interface and protocol of control unit borrowed from the company Výťahy ZEVA s. r. o. There was necessary to find suitable device for communication between control unit and server. Server is used for store and visualize actual data of elevator system. Server can set the parameters of the elevator remotely and demand actual configuration of elevator. User can watch historical data from server. The users use server after authentication with different competency. Users with an appropriate competency can set the notifications in the system, which can warn their in case of problem with elevator and reduce the reaction time and time to resolution of the problem.

Klíčová slova

REST API, Raspberry pi, C#, MS-SQL, Python, HTML, CSS, Angular, Typescript, Bootstrap, Vzdálená správa

Keywords

REST API, Raspberry pi, C#, MS-SQL, Python, HTML, CSS, Angular, Typescript, Bootstrap, Remote lift management

Citace

WEISSER, Filip. *Vzdálená správa výtahu.* Brno, 2018. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Mgr. Roman Trchalík, Ph.D.

Vzdálená správa výtahu

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Mgr. Romana Trchalíka Ph.D. Další informace mi poskytli pracovníci firmy Výťahy ZEVA s.r.o. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Filip Weisser
7. května 2018

Poděkování

Panu Mgr. Romanu Trchalíkovi, Ph.D., bych rád poděkoval za poskytnuté konzultace a veškerou pomoc při tvoření této diplomové práce. Dále bych chtěl poděkovat firmě Výťahy ZEVA s. r. o. za zapůjčení řídicí jednotky a všem pracovníkům této firmy, kteří mi poskytli potřebné materiály, odpovídali na mé otázky a pomohli mi vyřešit situace, se kterými jsem se setkal při práci s řídicí jednotkou.

Obsah

Kapitola 1

Úvod

Od doby, kdy byly výtahy pro zvedání těžkých břemen ovládány lidskou či zvířecí silou uplynulo mnoho let. V dnešní době provázené velmi rychlým technologickým vývojem a modernizací bylo dosaženo u výtahů velkých pokroků. Výtahy jsou důležitou součástí vysokých budov včetně těch inteligentních a najdeme je také v některých moderních rodinných domech, kde pomáhají lidem s postižením. Výtahy dnes disponují moderními mechanizmy jako je rekuperace energie, strojové učení pro odhadnutí nejvytíženějšího patra a neslouží jen pro přepravu pasažérů a těžkých břemen, ale také k zabezpečení budov proti neoprávněnému vniknutí. Výtahy obsahují velké množství bezpečnostních prvků s identifikací chyby výtahu, která pomáhá pracovníkům opravujícím výtah najít a odstranit vzniklou závadu. Technikům by mohlo pomoci vědět informaci indikující závadu ještě před tím, než se dostaví na místo. Vedení firmy může mít také přehled o stavu a poruchovosti všech výtahů, což může ovlivnit jejich rozhodování při koupi součástek od různých dodavatelů.

I když jsou na trhu dostupná řešení, mnoho malých firem si je z důvodů vysoké pořizovací ceny nemohou dovolit. Proto je cílem práce vytvořit snadno použitelný a co nejvíce cenově dostupný systém, s kterým mohou pracovníci těžit z výhod popsaných výše. Pro cenovou dostupnost je potřeba najít co nejlevnější převodník, který bude mít požadované vlastnosti pro komunikaci s výtahem a serverem. Systém by měl obsahovat co nejvíce funkcí, které mají ve svých řešeních velké firmy využívající monitoring. Navržený systém bude umožňovat vzdáleně nahlížet na aktuální stavy výtahů, případně na historická data jednotlivých výtahů. V případě chyby může podle aktuálního stavu výtahu pracovník poznat, jaké pomůcky pro opravu vzniklé chyby potřebuje. Na místo se také může dostavit pouze člověk, který konkrétní chybě rozumí více a jejíž oprava mu bude trvat kratší dobu. Pro co nejrychlejší odezvu na vzniklý problém bude systém umožňovat uživatelům systému nastavení notifikací, které je upozorní ihned při výskytu problému. To může zredukovat čas opravy porouchaného výtahu. Na požadavek od zákazníka by mělo být možné některé hodnoty zařízení nastavovat ze systému vzdáleně. Může se jednat například o rychlost otevírání dveří či intenzitu světla v kabině výtahu. Aby vzdálené nastavování parametrů nebylo zneužito, je nutné pro server navrhnout bezpečnostní mechanizmy pro komunikaci s převodníkem. Pro využití systému bude nutné přihlášení. Uživatelé budou mít různá oprávnění. S nejnižším oprávněním bude možné pouze nahlížet na stav výtahu. Protože systém může používat více firem, je nutné zajistit, aby po přihlášení uživatelé viděli pouze výtahy příslušné firmy.

V kapitole 2 se nachází popis dnešního stavu výtahových systémů. Na úvod kapitoly je popsáno, kde jsou výtahy integrovány. Dále je stručně popsána podoba výtahového systému. Následuje popis technologií, které můžeme u výtahových systémů v dnešní době

vidět. Poskytované služby budou porovnány napříč různými firmami. V kapitole 3 jsou analyzovány požadavky na systém vzdálené diagnostiky. Nachází se zde popis vlastností, které musí splňovat převodník, serverová část a celý systém. V této kapitole je představena řídicí jednotka, zapůjčená firmou Výťahy Zeva, a to včetně popisu komunikačního protokolu. Kapitola 4 obsahuje popis vhodného HW k implementaci systému. Dále je uveden návrh zabezpečení a architektury systému i serveru. V kapitole 5 je uveden popis implementace systému. Je stručně představena REST API architektura a testovací zařízení. Popis způsobu realizace převodníku a serverové části je uveden v oddělených kapitolách. V posledních kapitolách je nastíněn způsob testování systému a je ukázána finální podoba sytému.

Tato práce navazuje na semestrální projekt. V semestrálním projektu byla vypracována kapitola 2 o shrnutí současného stavu. Dále byly převzaty kapitoly: 3 a 4 o analýze a návrhu systému. Některé pasáže posledně dvou zmíněných částí byly v diplomové práci upraveny.

O implementaci systému vzdálené diagnostiky bylo usilováno již v rámci bakalářské práce [19]. Z důvodu absence převodníku a vlastností, které byly firmou později dospecifikovány, nebylo možné systém nasadit do reálného provozu. Nasazení systému do reálného provozu a přidání dalších klíčových vlastností je řešeno v této diplomové práci. Protože by bylo přidání nově specifikovaných vlastností do bakalářské práce obtížné, je v diplomové práci proveden nový návrh systému vzdálené diagnostiky, na který navazuje implementace.

Kapitola 2

Shrnutí současného stavu

Primárním využitím výtahů je stále přeprava těžkých břemen. V dnešní době je ale využití výtahů širší. Začínají se objevovat velmi zajímavé koncepty, se kterými se v bližší či vzdálenější budoucnosti setkáme. Už v dnešní době existuje velké množství druhů výtahů, které lze zaintegrovat do různých typů řešení podle potřeby. V této kapitole bude shrnuto, do jakých typů budov lze výtahy implementovat a bude představen systém výtahu v podobě, v jaké ho známe dnes. Dále je uveden popis situace vzdálené zprávy výtahu v dnešní době. V poslední řadě zde budou nastíněny nové technologie, jejichž rozšíření můžeme u výtahů očekávat v budoucnosti.

2.1 Integrace výtahů do existujících řešení

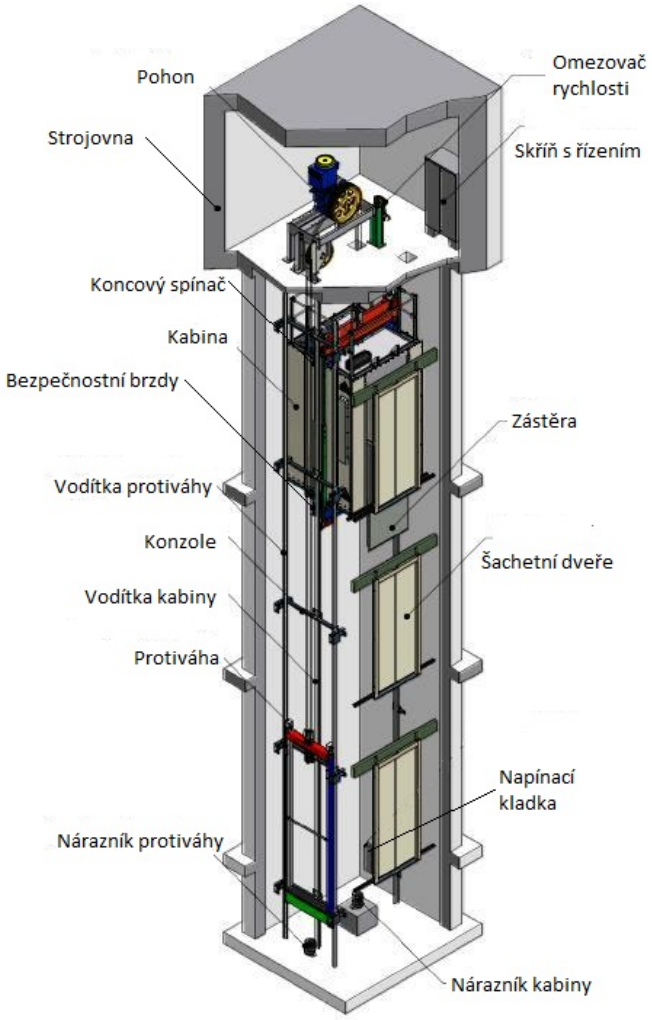
Výtahové systémy se v různých podobách integrují do všech typů budov, ať už se jedná o rodinné domy, různá firemní sídla, bytové domy, hotely, pensiony, inteligentní budovy a další budovy, kde je požadavek na rozšíření mobility a zvýšení kvality života. Mezi hlavní důvody k integraci výtahů do budov patří zvýšení kvality života a navýšení hodnoty budovy. Mezi další výhody patří přístupnější bydlení zejména pro starší a tělesně postižené osoby [11]. V dnešní době lze výtahy instalovat také kvůli bezpečnosti. Mohou sloužit například k znemožnění přístupu osob do určitých pater budovy či k evakuaci osob.

Výtahy můžeme rozdělit do dvou hlavních skupin, které jsou pojmenované podle typu použití. Jedná se o výtahy **nákladní** a výtahy **osobní**. Dále lze jako typ výtahu považovat pohyblivé schody neboli eskalátory. V této podkapitole budou uvedeny nosnosti výtahů, které nabízí ThysenKrupp jedna z největších výtahových firem světa [13]. Ostatní firmy mají ve svém sortimentu přibližně stejné nosnosti výtahů. Nákladní výtahy mají nosnost přibližně od 500 kg až do 15 000 kg. Spisové výtahy, které lze také považovat za druh nákladních výtahů mohou mít nosnost už od 50 kilogramů a slouží pro přepravu malých nákladů v administrativních sídlech apod. Jak už název vypovídá, nákladní výtahy slouží pro přepravu materiálů. Jsou použity zejména v průmyslových objektech, skladech a v podobných typech budov [10]. Osobní výtahy mají nižší nosnost, než výtahy nákladní, mají však vyšší rychlost a větší maximální výšku zdvihu. Nosnost osobních výtahů se pohybuje od 250 kg - 8 000 kg, jejich rychlost je přibližně 1 m/s - 20 m/s. Nejrychlejší výtah světa se nachází v Shangai Tower s rychlostí 20.5 m/s, což je v přepočtu 74 km/h [18]. U osobních výtahů převažují trakční typy pohonů. Vlastnosti výtahů jsou při instalaci přizpůsobeny konkrétním typům budov. Je nutné brát ohled na parametry jako je počet obyvatel v budově, výška budovy, velikost plochy pro šachetní otvor apod. Kromě větších budov můžeme výtahy čím

dál častěji vidět také v rodinných domech. V těchto případech jsou výtahy integrovány zejména pro osoby, které se do vyšších pater nedostanou bez pomoci. Dnešní moderní výtahy mohou přidávat budovám prvky inteligence, což je důvod, proč jsou důležitou součástí inteligentních budov.

2.2 Výtahový systém

Výtahový systém se skládá z mnoha částí. V této podkapitole budou pro představu o tom, jak výtah funguje popsány ty základní z nich. Popis se týká osobního trakčního výtahu, který je v budovách nejrozšířenější. Pro přehlednost lze vidět všechny komponenty na schématu níže.



Obrázek 2.1: Schéma výtahového systému. Převzato z [9].

Na obrázku je možné vidět nejdůležitější prvky výtahového systému. V horní části se nachází strojovna, ve které je umístěn pohon výtahu, omezovač rychlosti a skříň s řízením. V tomto schématu je vyobrazen trakční pohon, který má kabinu zavěšenou na soustavě lan. Pohyb kabiny je zajištěn navíjením lana na trakční kotouč výtahového stroje a může být ulehčen protiváhou. U hydraulických pohonů se strojovna nachází ve spodní části vý-

tahového systému. Pohyb kabiny zajišťuje elektricky poháněné čerpadlo, které dopravuje hydraulický olej do hydromotoru, který působí na klec. Omezovač rychlosti, nacházející se ve strojovně, kontroluje rychlost výtahu a v případě jejího překročení dojde k sepnutí bezpečnostních brzd a zastavení klece [9]. Tento mechanizmus hlídá jak výkyvy rychlosti, které většinou souvisí s poruchou, tak nekontrolovaný propad výtahové kabiny do šachetního otvoru v případě poškození lan. Poslední důležitou částí nacházející se ve strojovně je skříň s řízením. Výtahová skříň je pro tuto diplomovou práci stěžejní. Obsahuje komponenty pro řízení a kontrolu stavu výtahu. V dnešní době jsou pro účely řízení výtahu kvůli své kompaktnosti a malému odběru elektřiny používány mikroprocesory, které se objevily v prvním výtahu už v roce 1979 [2]. Mikroprocesor není omezen pouze řízením jednoho výtahu, což redukuje velikost skříně s řízením v budovách, kde je výtahů instalováno více. Mikroprocesor zodpovídá za vyhodnocování stavů všech senzorů a interakcí uživatele. Senzory poskytují data o stavu výtahu jako je: poloha výtahu, směr pohybu výtahu, zatížení výtahu, přivolání výtahu, chybovost výtahu apod. Jako výtahové kontroléry se v dnešní době používají PLC zařízení, které často obsahují jeden či více mikroprocesorů. PLC mohou být konfigurovány pro konkrétní výtah. Pro výtah lze nastavovat velké množství parametrů. Příkladem mohou být regulace plynulosti rozběhu a dojezdu výtahu. Výčet parametrů, které je možné nastavit v zapůjčené řídicí jednotce pro tuto diplomovou práci je uveden v příloze A. PLC kontroléry mohou obsahovat rozhraní pro zpřístupnění stavů. Rozhraní pro zpřístupnění informací o výtahu je využito v diplomové práci pro monitorování výtahu.

Pod strojovnou se nachází šachetní otvor, ve kterém se pohybuje kabina výtahu. Pro jízdu v šachetním otvoru jak kabiny, tak protiváhy jsou na stranách šachty připevněna vodítka. Ty jsou připevněny ke stěně šachty pomocí takzvaných konzolí [9]. Vodítka slouží k vedení kabiny výtahu a rámu protiváhy pomocí součástí připojených ke kabině výtahu a zapadajících do vodítek. Do rámu protiváhy lze podle potřeby přidávat závaží tak, aby byla co nejlépe vyvážena kabina výtahu. Ve spodní části se pro dojezd kabiny i protiváhy nachází nárazníky sloužící pro zmírnění dojezdu výtahu či protiváhy do spodního patra. Pro výtahy s menší rychlostí je často použit nárazník v podobě pružiny. Pro výtahy s vyššími rychlostmi je použit píst s olejem [9]. V každém patře se dále nachází šachetní dveře. Kabina výtahu disponuje dalšími dveřmi. Šachetní dveře mají mechanismus pro nouzové otevření dveří technikem v případě poruchy výtahu. Dveře je možné pomocí speciálního klíče mechanicky otevřít i v případě výpadku proudu. V šachetní části se pro monitorování stavu výtahu nachází řada senzorů. Může se jednat o například o senzory, které určují polohu kabiny výtahu. Tyto senzory jsou umístěny v každém patře. Další senzory polohy jsou umístěny těsně před každým patrem. Při příjezdu výtahu na tento senzor se výtah zpomalí a dojede plynulou jízdou do konce patra. Senzory se nenacházejí pouze v šachetním otvoru, ale také například v motoru výtahu. Senzory v motoru měří teplotu a při překročení určité hodnoty mikroprocesor zamezí dalšímu pohybu výtahové kabiny. Výčet chyb, které obsahuje zapůjčená řídicí jednotka, se nachází v příloze A. V horní části šachetního otvoru se nachází koncový spínač, který zajišťuje zastavení výtahu v krajní poloze horního patra.

2.3 Současný stav

V této podkapitole bude stručně shrnuto, jaké technologie se u výtahů používají v současné době. Jedná se o technologie, které některé firmy využívají již teď a jejichž rozšíření můžeme očekávat v budoucnu. Nachází se zde shrnutí a popis vzdálené diagnostiky. Na konci je shrnuto v přehledné tabulce využití všech zmíněných technologií jednotlivými firmami. Budou v ní zmíněny jak menší firmy působící v České republice, tak globální firmy působící kromě České republiky po celém světě.

2.3.1 Rádio Frekvenční Identifikace

RFID (Radio Frekvenční Identifikace) je moderní technologie identifikace objektů (v tomto případě lidí používajících výtah) pomocí radio-frekvenčních vln. RFID systém lze využít ve velkém množství odvětví a je široce využívaný v inteligentních budovách [21]. S implementací této technologie může pouze identifikovaná osoba použít výtah a zmáčknout přivolávací tlačítko v kabině. Tím lze zvýšit bezpečnost osob v budovách. Ty se mohou ve výtahu pohybovat bez starostí, že je může obtěžovat někdo, kdo k použití výtahu nemá přístup. Chránit lze také majetek, neboť instalace RFID sytému redukuje pravděpodobnost krádeží [15]. Bezpečnost v budovách lze zvýšit i tím, že kontrolér výtahu povolí identifikované osobě přivolat pouze patra, ke kterým má nastavený přístup. Při každé jízdě systém zaznamenává datum a číslo podlaží, kterého uživatel dosáhl. Tak lze při problému určit, kdo se v určitém čase nacházel na daném podlaží. Jméno či identifikátor se může objevit například na LCD displeji ve výtahu. Implementací RFID je možné redukovat počet jízd výtahu, protože výtah mohou používat pouze autorizované osoby. Tím lze zmenšit opotřebení výtahu a náklady na jeho provoz. Redukování počtu jízd může být velmi žádoucí například v nemocnici, kde je při naléhavých případech nutná okamžitá dostupnost výtahu. Protože personál nemocnice i veřejnost sdílí stejný výtah, mohou například v době návštěv nastat situace, kdy je výtah v době naléhavých případů nedostupný. Proto, aby se předešlo tomuto problému stačí, aby měl personál nemocnice platný tag. Po autorizaci se může pouze zaměstnanec nemocnice dostat na požadované patro.

Protože jsou jízdy zaznamenávány, lze použít učící se technologie. Pokud je výtah delší dobu nepoužíván, po uplynutí určité doby kabina výtahu sjede do vhodného podlaží. Vhodnost je určena díky frekventovanosti použití výtahu v dané hodině, případně dni.

RFID se skládá ze dvou hlavních komponentů, tagu a čtečky. Tag vlastní uživatel výtahu většinou ve formě karty, ve velikosti podobné kreditní kartě. Jsou v něm uloženy osobní informace uživatele a jeho práva. Uživatel může dosáhnout určitého patra pouze se specifickým oprávněním uloženém na kartě. Každému uživateli lze nastavit odlišná oprávnění. Nastavení karty uživatele probíhá v Systém Management centru. Tag a čtečka spolu komunikují pomocí RFID technologie. K autorizaci musí čtečka načíst tag, pomocí kterého identifikuje uživatele. Pokud je tag validní, čtečka odešle signál ke zpracování kontroléru. Poté může uživatel zvolit patro zmáčknutím tlačítka v kabině výtahu. Na displeji výtahu jsou po autorizaci uživatele vyobrazeny informace o uživateli včetně jeho oprávnění.

RFID kontrolér a kontrolér výtahu jsou odděleny, takže instalace nemá vliv na původní funkcionalitu výtahu. Identifikace neobtěžuje uživatele, neboť RFID autorizace probíhá bezdrátově na delší vzdálenost, a tak není nutné identifikační kartu vytahovat. Implementace RFID do výtahu je jednoduchá. Stačí pouze připojit vodič k panelu s tlačítky uvnitř výtahu a instalovat RFID kontrolér na každé podlaží k zamezení přivolávání výtahu neidentifikovaným osobám [21].

2.3.2 Rekuperace energie

V dnešní době se hledají metody, kterými lze ušetřit co nejvíce elektrické energie. Jednou z metod je rekuperace elektrické energie. Rekuperace je čím dál více rozšířenější. Lze ji vidět například v automobilovém průmyslu u hybridních aut, kde jsou v synergii optimálně využívány různé zdroje energie [14]. Komponenty používané při rekuperaci, jako jsou například superkapacitory, v posledních letech přitáhly pozornost mnoha firem. Rekuperaci lze využít kromě automobilového průmyslu i v dalších odvětvích včetně výtahových systémů. Rekuperační jednotku je možné nainstalovat téměř na každý druh výtahu a její efektivita se odvíjí od vytížení výtahu [5].

Rekuperace je proces přeměny potenciální energie zpět na využitelnou elektrickou energii při brzdění [16]. Elektrická energie je buď vrácena do napájecí sítě nebo uložena do akumulátorů (superkapacitorů), kde je posléze využita k napájení vnitřní elektroniky výtahu. Kromě snížení spotřeby elektrické energie jsou dalšími výhodami rekuperace eliminace tepelných ztrát, ochrana životního prostředí, tichý chod výtahu a jeho vysoká přesnost zastavení.

2.3.3 Solární napájení

Stejně jako v předchozí podkapitole je použití této technologie zaměřeno na ušetření provozních nákladů výtahu. Společnosti se snaží najít způsob, jak nabízet co nejvíce energeticky úsporné produkty. Sluneční energii lze pomocí fotovoltaického jevu přeměnit na energii elektrickou. Této skutečnosti využívají firmy, které nabízejí zapojení solárních panelů pro různé druhy využití. Některé výtahové společnosti poskytují zapojení solárních panelů a nabízejí takzvané solární výtahy [12]. Tyto výtahy jsou napájeny převážně nebo úplně pomocí solární energie a jsou často kombinovány s již zmíněnou rekuperační jednotkou.

Výtah s touto technologií lze provozovat pomocí solární energie s kombinací energie z elektrické sítě podle konfigurace a dostupnosti slunečního světla. Získaná energie může být použita okamžitě nebo uložená do akumulátorů. Pomocí nabitých akumulátorů lze výtah provozovat i při výpadku elektrického proudu. V dnešní době poskytují solární výtahy převážně zahraniční společnosti, které působí také na českém trhu. Počty konfigurací však nejsou velké a převážně je u výtahů k šetření elektrické energie instalována pouze rekuperační jednotka.

2.3.4 Vzdálená diagnostika

Z technologií a systémů popsaných v této kapitole je zjištění současného stavu vzdálené diagnostiky pro tuto diplomovou práci nejdůležitější. Někteří domovníci si mohou stěžovat na závady, které nemají s poruchou nic společného. To může být bez vzdálené diagnostiky značně frustrující. Provoz výtahů či eskalátorů se stává stále více přístupnější pro veřejnost a to není pro výtahové společnosti pozitivní. Pokud má výtahový systém nainstalovány prvky pro vzdálenou diagnostiku, je v reálném čase monitorován jeho stav a jsou sledovány systémové funkce výtahového systému. Zákazníci, kteří mají k systému přístup se proto mohou informovat o stavu svého výtahu. Může se jednat o informace jako je pohyb výtahu, vytíženost výtahu, aktuální patro výtahu, informace o otevřených dveří výtahu apod. Pokud se u výtahu vyskytne závada, mohou se uživatelé s přístupem podívat na typ poruchy a technici mohou odhadnout její příčinu. Technikům, spravujícím výtah, je navíc odeslána notifikace formou emailu či SMS zprávy a tak ví o vzniku závady okamžitě. To je pro společnosti výhodou, neboť se mohou na místo dostat velice rychle a zkrátit tak dobu,

po kterou je výtah nefunkční. Protože jsou technici dopředu informováni o chybě, mohou si vzít na místo potřebné vybavení. Tím je také redukována doba, po kterou je výtah nefunkční, neboť v případě chybějící součástky je technik nucen obstarat si ji a odjet od místa poruchy. K systému lze nastavit oprávnění přístupu a stav svého výtahu mohou sledovat například domovníci. Ze vzdálené diagnostiky mají přínos nejen společnosti, ale i zákazníci. Nevýhodou je počáteční pořizovací cena, a to hlavně u výtahů, které nemají pro diagnostiku připravenou řídicí jednotku.

Možnosti vzdálené správy a monitoringu všech zařízení byly v posledních letech rozšířeny, čehož si začali všímat i výrobci výtahů. V nedávné minulosti bylo nevýhodou, že byl monitoring vyráběn primárně pro nové výtahové stroje. Proto rozšířenost vzdálené správy a monitoringu nebyla velká. Nyní se instalace vzdáleného monitoringu netýkají jen nových výtahů a vznikají například obecná PLC řešení a balíky [20]. Jsou dostupná řešení zaměřující se pouze na určité části výtahu, například monitoring výtahových dveří. Malé firmy si vzdálenou diagnostiku z důvodu pořizovací ceny či vysokého paušálu nemohou dovolit. Vzdálenou diagnostikou kromě pár výjimek disponují pouze velké společnosti, které vlastní svou implementaci. Jedná se o společnosti OTIS, ThyssenKrupp a další. Proto je cílem diplomové práce nabídnout řešení i pro menší společnosti, které nedisponují velkými finančními prostředky.

2.3.5 Srovnání jednotlivých firem

Tato podkapitola obsahuje tabulku, ve které lze vidět, jaké firmy technologie popsané výše poskytují. Protože je výtahových firem velký počet, srovnání je zaměřeno na firmy působící v Moravskoslezském kraji. I přesto se v tabulce objevují globální firmy, které mají výtahy po celém světě. Globální firmy působící na Moravě a ve Slezsku jsou například Otis, ThysenKrupp, Kone a Schindler. U některých firem nelze informaci o tom, že danou technologii nabízí vyhledat. Pokud pracovníci těchto firem neodpověděli na dotaz prostřednictvím emailu, je to v tabulce znázorněno otazníkem. U firem, které technologii nabízí, se nachází symbol odškrtnutí. Firmy, které technologii nenabízí, mají pole prázdné.

	RFID	Rekuperace	Solární napájení
Otis	✓	✓	✓
ThysenKrupp	✓	✓	✓
Kone	✓	✓	✓
Betacontrol	✓	✓	
Schindler	✓	✓	✓
Liftcomp	?	?	?
Liftmont	?	?	?
Novalift			
Výtahy Ostrava	?	?	?
LiftComponents	?	✓	?
Lift Servis			
Cenok Výtahy	?	✓	?
Msvytahy	?	?	?
AZ Výtahy	✓	?	?
Konekta	?	?	?

Tabulka 2.1: Tabulka dostupnosti technologií u jednotlivých firem

	Vzdálená diagnostika
Otis	✓
ThysenKrupp	✓
Kone	✓
Betacontrol	✓
Schindler	✓
Liftcomp	?
Liftmont	?
Novalift	
Výtahy Ostrava	?
LiftComponents	
Lift Servis	✓
Cenok Výtahy	?
Msvytahy	?
AZ Výtahy	?
Konekta	?

Tabulka 2.2: Tabulka dostupnosti vzdálené diagnostiky

Kapitola 3

Analýza

Před tím, než bude systém implementován, je nutné analyzovat všechny potřebné vlastnosti, které by měl splňovat. Z předešlé kapitoly vyplývá, že je pro malé zákazníky, kteří stále nevyužívají systém vzdálené diagnostiky, důležitá cena. V této kapitole se nachází analýza dalších skutečností potřebných k implementaci kvalitního softwaru. V první podkapitole jsou shrnuty hlavní požadavky na budoucí systém a to jak z pohledu hardwaru, tak z pohledu softwaru. Na konci kapitoly se nachází rozbor protokolu, kterým bude komunikovat zapůjčená řídicí jednotka se systémem.

3.1 Požadavky na systém

Navržený systém bude obsahovat dvě hlavní části: server a zařízení komunikující se serverem. V této diplomové práci je zařízení simulováno testovacím modulem. Protože zařízení neobsahuje prostředky pro komunikaci se serverem, musí se pro tyto účely najít a implementovat vhodný převodník. Celý navržený systém včetně převodníku musí být cenově dostupný. Systém by měl přehledně zobrazovat aktuální stavy zařízení. To souvisí s dobře použitelným a příjemným uživatelským rozhraním. Veškeré zařízení musí jít jednoduše vyhledávat a filtrovat podle rozdílných kritérií. Protože systém může využívat více firem je nutné zajistit, aby uživatelé viděli pouze zařízení, které spravují. Návrh by měl být co nejvíce obecný a modulární pro jednoduché přidání či změnu vlastností systému. Zařízení mají svá nastavení. Po požadavku uživatele by měl výtah zaslat kompletní konfiguraci a přehledně ji zobrazit. Systém by měl zajistit, aby bylo možné všechny informace uvedené v konfiguraci vzdáleně nastavovat. Protože systém využívá více lidí a poskytuje informace, které by neměly být přístupné všem, je nutné aby disponoval možností autentizace. Autentizovat se do systému by mělo být možné s více úrovněmi oprávnění. Jaké druhy uživatelských rolí požaduje firma, pro kterou je tato diplomová práce určena, je možné vidět v podkapitole 3.1.2. Systém by měl umožňovat vzdáleně nastavovat parametry a poskytovat informace o zařízeních firem. Proto je nutné, aby kromě autentizace byla určitým způsobem zabezpečena komunikace mezi převodníkem a serverem. V případě vzniklé chyby by se pracovníci firem měli co nejrychleji dozvědět o jejím vzniku. Systém by měl ihned při vytvoření chyby informovat technika. Tomu musí být umožněno nastavit si podle potřeby různé druhy notifikací.

3.1.1 Hardware

Pro komunikaci se serverem je řídicí jednotku výtahu nutné připojit na vhodné zařízení. Toto zařízení by mělo mít co nejvíce možností pro připojení k internetu. Jelikož řídicí jednotka komunikuje přes sériové rozhraní, musí hardware umožnit libovolným způsobem přijímat sériovou komunikaci. Zařízení musí být dostatečně rychlé a nabízet konektivitu pro připojení dalších částí, které mohou diagnostický systém v budoucnosti rozšiřovat.

3.1.2 Software

Jednotlivé softwarové požadavky byly popsány v úvodu této kapitoly a jejich výčet je možné vidět níže. Druhy účtů pro přihlášení jsou detailněji popsány v podkapitole autentizace.

Požadavky na systém:

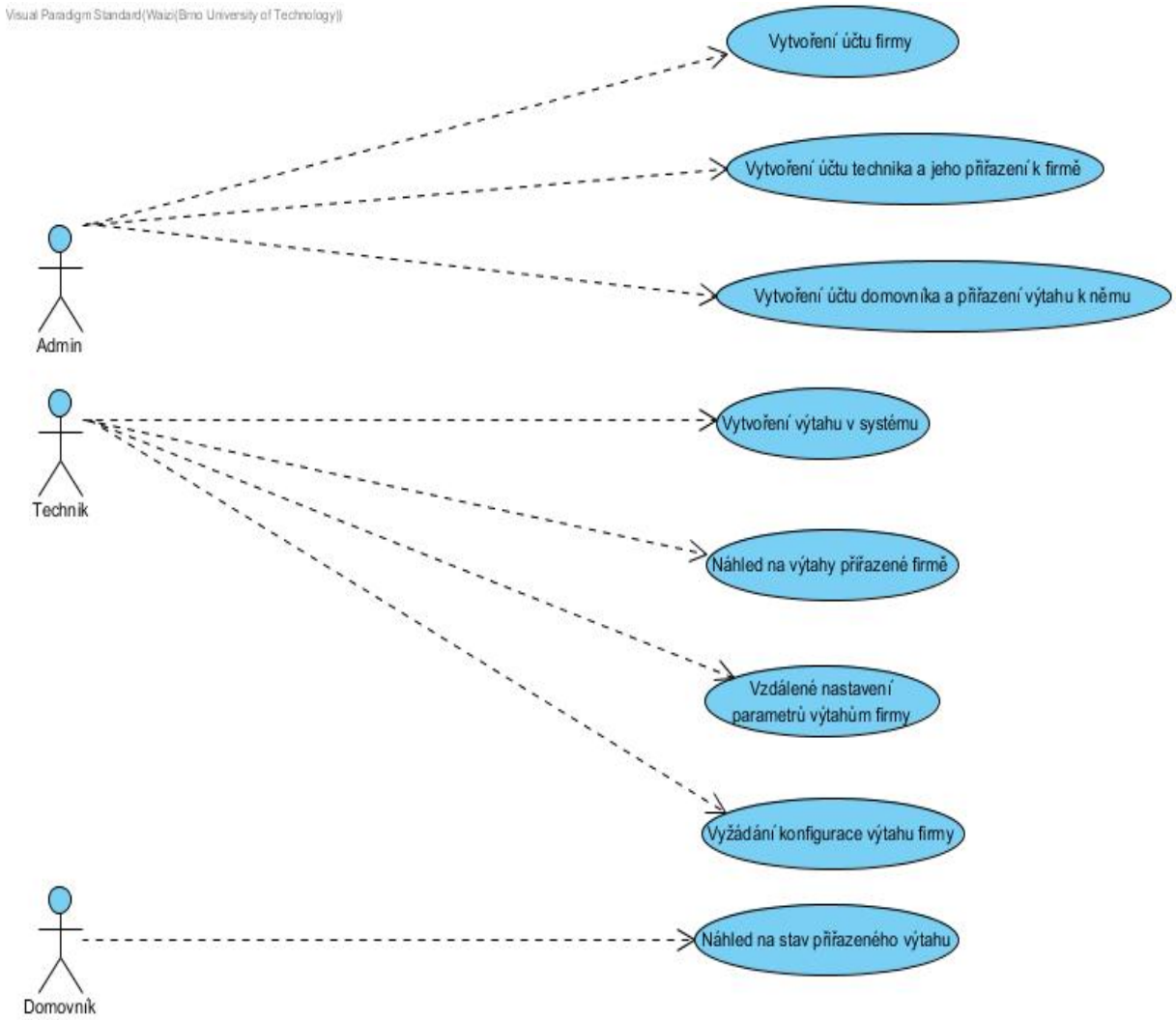
- zobrazení aktuálních informací o výtahu s možností filtrování,
- autentizace s více druhy oprávnění,
- obecnost návrhu,
- zaslání celé konfigurace výtahu na požadavek uživatele,
- vzdálené nastavení parametrů konfigurace uživatelem,
- notifikace uživatelů při chybě a možnost nastavení typu notifikace,
- bezpečnost,
- modulárnost.

Zobrazení aktuálních informací o výtahu s možností filtrování

Systém bude zobrazovat aktuální informace o stavech různých výtahů firem. Je potřeba počítat s tím, že firmy mohou mít stovky či tisíce výtahů a bez dobrého uživatelského rozhraní může být systém špatně použitelný a nepřehledný. Je nutné zajistit, aby uživatel v krátkém čase našel konkrétní výtah a potřebné informace. Proto by měl systém umožnit filtrovat podle více kritérií a mít dobře navržené uživatelské rozhraní.

Autentizace

Firma Výťahy Zeva definovala tři různé role a jejich oprávnění, které je možné vidět na diagramu užití níže. První rolí s nejmenším oprávněním je domovník. Domovník bude mít povoleno pouze nahlížet na stavy výtahu, které mu byly přiděleny. V detailech zařízení mu budou poskytnuty pouze omezené informace a možnosti. Další rolí v systému je technik, který může nahlížet na stav všech výtahů firmy, vyžádat si jejich konfiguraci a vzdáleně nastavovat parametry. Technik může vytvářet výtahy, které jsou automaticky přiřazeny jeho firmě. Poslední rolí je administrátor sloužící pro vytváření účtů v systému. Administrátor má povoleno vytvářet účet domovníka a přidělit mu konkrétní výtah. Další zodpovědností administrátora je vytváření účtu techniků a jejich přiřazení ke konkrétní firmě. Role a jejich oprávnění jsou uvedeny v diagramu níže.



Obrázek 3.1: Diagram případu užití firmy Výťahy Zeva

Obecnost návrhu

Systém bude využívat více firem. Různé firmy mohou mít různé typy chyb. Také se mohou lišit typy zpráv a protokolem řídicí jednotky. Zpracování zpráv může probíhat odlišným způsobem. Návrh systému musí být natolik obecný, aby při požadavku dalších firem nebyl zásah do systému fatální.

Zaslání celé konfigurace výtahu na požadavek uživatele

Výtahy uchovávají informace o svých nastavených parametrech. Může se jednat například o nastavení rychlosti revizní jízdy či nastavení doby, po kterou budou dveře ve stanici otevřené. Pomocí systému by měl mít revizní technik možnost informovat se o nastavených parametrech konkrétního výtahu. Protože jsou všechny informace přenášeny v binární formě, měly by se jednotlivým hodnotám pro přehlednost přiřadit vhodné názvy.

Vzdálené nastavení parametrů konfigurace uživatelem

Technikům by jistě ulehčilo práci, kdyby mohli výše popsanou konfiguraci výtahu nastavovat vzdáleně. Tak jako u žádosti konfigurace výtahu by veškeré hodnoty měly mít vhodný popis. Nastavovat by se měly konkrétní hodnoty (ano/ne, číselné hodnoty), které budou převedeny na konkrétní binární hodnoty registrů pro výtah.

Notifikace uživatelů při chybě a možnost nastavení typu notifikace

Pro okamžité informování technika o vzniklé závadě by měl systém obsahovat možnost nastavení notifikace technikem. Technik by měl mít možnost výběru z více typů notifikací a jejich jednoduchou konfiguraci. Může upřednostňovat například emailovou notifikaci či zvolit SMS notifikaci. Aby byl technik informován jen o určitých typech zpráv, mělo by jít nastavit notifikace také podle úrovně závažnosti poruchy.

Bezpečnost

Informace o výtazích by měly být přístupné pouze autentizovaným uživatelům. Komunikace mezi převodníkem a serverem probíhá na veřejném internetu a kdokoliv si může odchytnout jednotlivé pakety. Dalším možným nebezpečím je možnost vzdáleného nastavení parametrů výtahu neautorizovanou osobou. Komunikaci po internetu je z těchto důvodů nutné zabezpečit vhodným způsobem.

Modulárnost

Navržený systém by měl umožňovat přidání dalších komponentů k rozšíření jeho funkčnosti. K tomu je potřeba, aby byl systém co nejvíce modulární a rozšiřitelný. Mezi vhodné kandidáty pro rozšíření může patřit například systém RFID, popsaný v podkapitole 2.3.1. Pokud by byla v systému instalována technologie rekuperace energie či napájení ze solárního panelu popsaných v podkapitolách 2.3.2 a 2.3.3, bylo by možné systémem sledovat i stav těchto komponent. Systém by mohl zobrazovat informace o zařízeních, jako je například množství vyrobené energie či množství vrácené energie do sítě rekuperací.

3.2 Řídicí jednotka

Od firmy Výťahy Zeva byl zapůjčen testovací kit, na kterém se nachází řídicí jednotka výtahu. Na zapůjčeném modulu se nachází komponenty, pomocí kterých lze simulovat provoz výtahu. S tlačítky na řídicí jednotce lze simulovat přivolání výtahu. Kolečkem na testovacím

kitu jeho jízdu. Spínače SKRD, SKRH slouží pro simulaci stavu, kdy výtah dojede do prvního či posledního patra. To znamená, že je sepnuto čidlo nacházející se reálně v krajních polohách šachty výtahu. Na testovacím kitu lze některé chyby navodit pomocí spínačů.

3.2.1 Protokol řídicí jednotky

Pro implementaci systému vzdálené diagnostiky je nutné znát způsob, jakým řídicí jednotka výtahu komunikuje. V této sekci bude popsána struktura všech zpráv, kterými zapůjčená řídicí jednotka disponuje. Dále bude popsáno, kdy se jednotlivé zprávy posílají a jak probíhá komunikace s řídicí jednotkou.

Protokol výtahu obsahuje pět typů zpráv, které je možné vidět níže. Příklady popisu jednotlivých zpráv jsou uvedeny v ASCII formátu hexadecimální soustavy. Veškeré zprávy mají fixní velikost 22 bajtů. Ve všech zprávách je na prvních dvou bajtech žlutě označená hodnota - ID domu. Následující bajt označuje ID výtahu v domě pro případ, že je v budově výtahů více. Konkrétní výtah proto lze identifikovat pomocí třech bajtů. Čtvrtý červeně zvýrazněný bajt označuje typ zpráv odesílaných řídicí jednotkou. Na pátém bajtu je upřesňující informace o dané zprávě, jejíž vysvětlení je uvedeno v popisu konkrétních zpráv. Hodnota XX na posledním bajtu všech zpráv nezastupuje konkrétní hodnotu, ale určuje, že je na tomto místě kontrolní součet. Kontrolní součet zpráv se provádí XORem. XOR probíhá postupně od prvního bitu zleva až po bit poslední.

Zpráva o události

Zpráva o události je identifikovaná hodnotou **00** a informuje o vzniku poruchy. Zpráva se odesílá buď v případě poruchy, anebo pokud se výtah dostane z poruchového stavu opět do normálního chodu. Hodnota na pátém bajtu udává typ závady nebo informaci o tom, že závada skončila. E5 udává překročení doby jízdy z jedné stanice do druhé. Pokud by se v typu závady vyskytovala hodnota "BB", znamená to, že výtah přešel z chybového do funkčního stavu. Veškerý výčet chyb řídicí jednotky firmy Zeva je možné vidět v příloze A. Fialově označená hodnota na 21 bajtu udává patro, na kterém se chyba vyskytla.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
0A	02	02	00	E5	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	05	XX

Tabulka 3.1: Zpráva o události

Zpráva s počítadly

Zpráva s počítadly identifikovaná hodnotou **02** na čtvrtém bajtu je odeslána při každém rozjezdu výtahu. Má informativní účel. Pátý bajt udává patro, z kterého se výtah rozjel, a pro které se některé informace vztahují. V případě uvedeném níže se jedná o počítadlo pátého poschodí. Dalších 12 bajtů, které jsou vyznačeny zeleně, má informativní účel. První 4 bajty udávají celkový počet změn polohy výtahu. Další 4 bajty celkový počet změn ohybu lana výtahu. Na rozdíl od změny polohy výtahu se počet ohybu lana nezvětšuje, pokud výtah nezmění směr jízdy. Bajty 14-17 se vztahují k danému poschodí (zde k pátému) a udávají celkovou hodnotu počítadla daného poschodí. To znamená, kolikrát se výtah v daném poschodí nacházel. Tu samou informaci, ale pouze pro měsíční období, udávají bajty 20 a 21. Bajty 18 a 19 jsou rezervní.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
0A	02	02	02	05	00	00	00	01	00	00	00	02	00	00	00	03	FF	FF	00	01	XX

Tabulka 3.2: Zpráva s počítadly

Testovací zpráva

Testovací zpráva je identifikovaná hodnotou **0A0A** na čtvrtém a pátém bajtu. Zpráva je posílána řídicí jednotkou periodicky každých 30 sekund. Hodnota na 20-tém bajtu zvýrazněna zeleně udává, jestli je daný výtah v provozu či ne. Pokud je výtah v poruchovém stavu, obsahuje tento bajt identifikátor konkrétní chyby. V opačném případě obsahuje hodnotu BB. Hodnota bajtu 21 označuje, v jakém patře se výtah nacházel při odeslání zprávy. Kromě obvyklé komunikace slouží testovací zpráva pro začátek komunikace a potvrzování přijatých zpráv při vzdáleném nastavování parametrů. Při potvrzování přijetí konfiguračních registrů řídicí jednotkou je testovací zpráva identifikována hodnotou **0B0A**.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
0A	02	02	0A	0A	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	BB	05	XX

Tabulka 3.3: Testovací zpráva

Potvrzovací zpráva

K potvrzení přijetí a zpracování zprávy na straně serveru se v protokolu nachází potvrzovací zpráva. Kromě zprávy sloužící k vzdálenému nastavování parametrů se jedná o jedinou zprávu, která je v protokolu posílána směrem od serveru k výtahu. Je identifikována hodnotou **0A** na čtvrtém bajtu a hodnotou na pátém bajtu, která je nastavována podle kontextu. Hodnota na pátém bajtu nabývá hodnot 0B pro potvrzení testovací zprávy a 0C pro potvrzení ostatních zpráv popsaných výše. Další hodnoty, kterých může pátý bajt potvrzovací zprávy nabývat, jsou uvedeny při popisu konfigurační zprávy a vzdáleného nastavení či přijetí parametrů.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
0A	02	02	0A	0B	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	XX

Tabulka 3.4: Potvrzovací zpráva

Konfigurační zpráva

Konfigurační zpráva slouží jak pro vzdálené nastavení konfiguračních parametrů řídicí jednotce, tak pro zaslání konfigurace na stranu serveru. Řídicí jednotka firmy Výťahy Zeva obsahuje jedno-bajtové registry B0-BF a dvou-bajtové registry D0-DB, kde je konfigurace uložena. Žádost o konfiguraci i její vzdálené nastavení probíhá vždy pro všechny parametry, proto je nutné posílat dvě zprávy. Zpráva je identifikována hodnotou **01**. Pátý bajt udává, jestli jsou nastavovány či přijímány registry Bx či Dx. Výčet všech parametrů, kterými řídicí jednotka disponuje, je možné vidět v příloze A. Volné bajty jsou ponechány pro případnou rezervu v případě, že se do systému přidají další parametry.

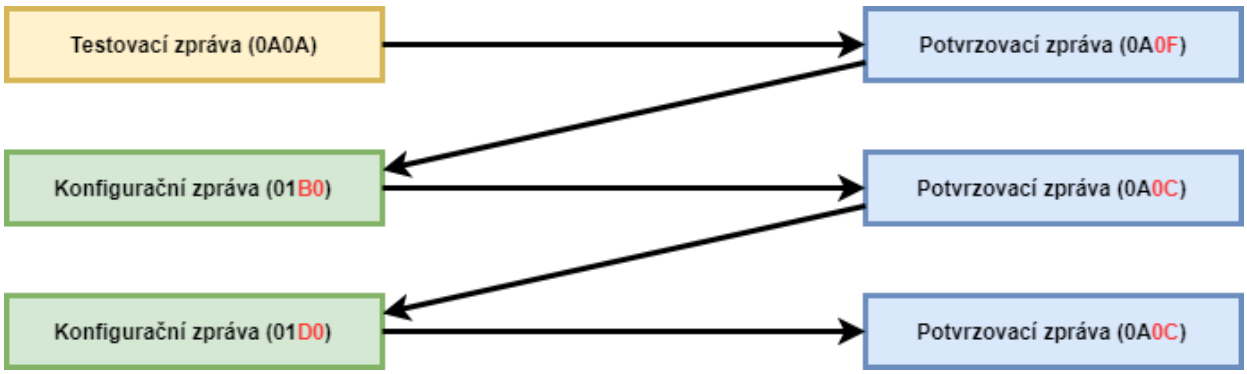
Pro požadavek na zaslání konfigurace je nutné na testovací zprávu řídicí jednotky odpovědět potvrzovací zprávou s hodnotou 0F na pátém bajtu. Po přijetí první částí konfigurace je nutné odpovědět potvrzovací zprávou s hodnotou 0C. Ukázka celé komunikace mezi převodníkem a řídicí jednotkou při zasílání vzdálených parametrů je na obrázku 3.2.

Pro vzdálené nastavení parametrů je nutné po přijetí testovací zprávy zaslat první konfigurační zprávu s registry B0-BF. Řídicí jednotka po přijetí této zprávy ihned odešle testovací zprávu, čímž potvrzuje přijetí. Po přijetí druhé testovací zprávy je možné ze strany serveru poslat druhou konfigurační zprávu s hodnotami registrů D0-DB. Řídicí jednotka potvrzuje přijetí vždy zasláním testovací zprávy s hodnotou 0B0A. Příklad komunikace je nastíněn v diagramu 3.3.

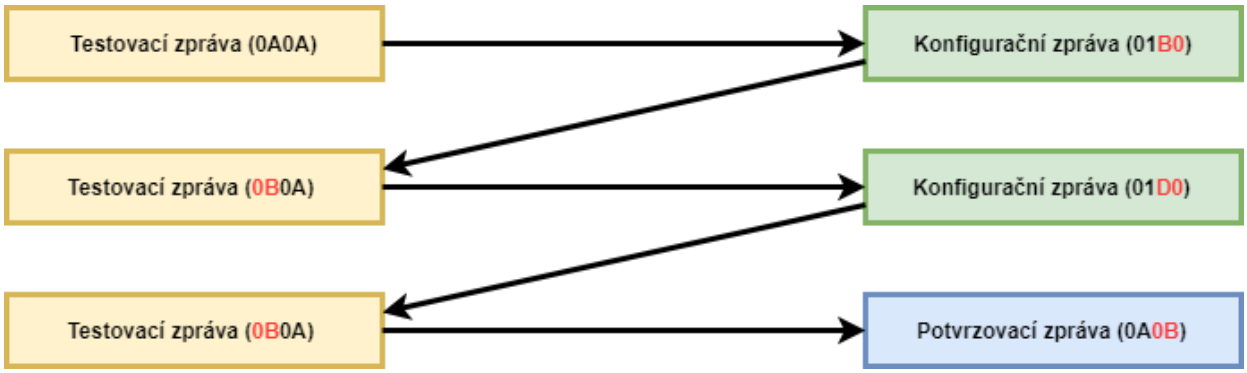
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22
0A	02	02	01	B0	10	10	00	10	01	00	00	01	10	11	10	01	00	00	01	FF	XX

Tabulka 3.5: Konfigurační zpráva

V následujících diagramech jsou pro lepší zřetelnost barevně odděleny typy zpráv. Pro zaslání konfigurace řídicí jednotky je nutné čekat na testovací zprávu. Jak je z popisu výše patrné, maximální doba čekání je 30 sekund. Po přijetí testovací zprávy je zahájeno zaslání konfigurace potvrzovací zprávou, která má hodnotu "0F"na pátém bajtu. Tato skutečnost je v diagramu zvýrazněna červenou barvou. Řídicí jednotka ihned odpovídá první konfigurační zprávou s registry B0-BF. Tu je nutné potvrdit potvrzovací zprávou, nyní však s hodnotou "0C"na pátém bajtu. Stejnou zprávou se potvrzuje také následující konfigurační zpráva s hodnotami registrů D0-DB. Ta je zaslána řídicí jednotkou ihned po potvrzení. Velmi podobným způsobem probíhá také vzdálené nastavení parametrů, které je nastíněno na následujícím obrázku. Vzdálené nastavení však započne tím, že se ihned po přijetí testovací zprávy pošle první konfigurační zpráva s hodnotami registrů, které požadujeme nastavit.



Obrázek 3.2: Komunikace mezi řídící jednotkou a serverem při vzdáleném zaslání konfigurace.



Obrázek 3.3: Komunikace mezi řídící jednotkou a serverem při vzdáleném nastavení parametrů.

Kapitola 4

Návrh

V této kapitole se vyskytuje popis návrhu částí, které se v dalších fázích projektu použijí k implementaci a sestavení systému pro vzdálenou diagnostiku. Ke komunikaci mezi řídicí jednotkou a serverem je potřeba zvolit vhodné zařízení a k zabezpečení této komunikace vhodné bezpečnostní prvky. V této kapitole je ukázáno, jak by měla vypadat architektura systému, a jaké komponenty bude obsahovat. Pro přehlednost je znázorněna komunikace mezi řídicí jednotkou a serverem diagramem. V kapitole architektura serveru je základní návrh pro budoucí implementaci této části. Vyskytuje se zde návrh databáze, který je možno vidět na ER diagramu a základní balíčky, které se budou v systému vyskytovat. V kapitole modularita systému se nachází návrh, který může být platný v případě, že bychom chtěli rozšířit funkčnost diagnostiky a další komponenty.

4.1 Hardware

Jako hardwarové řešení je možné vybrat některý z minipočítačů. Většina z nich má velký počet portů a různých připojitelných modulů. Všechny mají také výkon, který je dostatečný pro komunikaci s řídicí jednotkou a serverem, případně k rozšíření o další moduly. Všechny minipočítače jsou cenově dostupné a mají nízkou spotřebu energie.

Ze všech dostupných řešení byl pro implementaci převodníku zvolen minipočítač **Raspberry pi**. Tento minipočítač je na trhu dlouho a je dostupný ve více verzích. Pro vývojové účely byla zvolena poslední nejvýkonnější verze, která může být při instalaci do reálného výtahu vyměněna za levnější. Raspberry pi byl vybrán, neboť poskytuje všechny potřebné vlastnosti ke komunikaci mezi řídicí jednotkou a serverem. Nemá vlastní paměť, nicméně obsahuje slot pro SD kartu, na kterou je uložen operační systém. Pro komunikaci se serverem není velká interní paměť potřeba. SD karty mají v případě rozšíření o jiné moduly dostatečnou kapacitu. Raspberry pi má dobrý výkon a obsahuje dostačující počet konektorů pro případné rozšíření o další komponenty. Protože je na trhu dlouhou dobu, má z dostupných řešení největší velikost komunity, velké množství návodů a je dobře dokumentovaný. Ke komunikaci se serverovou částí je potřeba dostupnost internetu. Raspberry pi nabízí možnosti připojení k internetu pomocí Ethernetu či WiFi. Ethernet i WiFi jsou často dostupné ve strojovně, kde se nachází řídicí jednotka. Pokud by internet dostupný nebyl, lze použít modul pro GSM/GPRS komunikaci. S tímto řešením by bylo potřeba dokoupit platnou SIM kartu. Pro sériovou komunikaci s řídicí jednotkou bude použit USB to TTL kabel, pracující na 5V napětí. Pro implementaci převodníku bude použit jazyk Python a operační systém Raspbian, na kterém už je předinstalována většina potřebného softwaru.

4.2 Bezpečnost

Způsobů zabezpečení je v dnešní době celá řada. Pro zabezpečení komunikace bylo zváženo použití **VPN** technologie, zabezpečené **REST API (HTTPS)** a protokolu **SSH**. Protokoly mají velmi podobné mechanismy zabezpečení a zajišťují jak šifrování, tak identifikaci. Velké množství VPN implementací využívá pro zabezpečení IPSEC či SSL/TLS protokol. SSL/TLS protokol používá k zabezpečení také REST API. TLS je následovník SSL protokolu a opravuje bezpečnostní chyby předchozích verzí. Pracuje na transportní vrstvě, SSH na vrstvě aplikační [17]. Protokol SSH je hojně používaný administrátory systému k přístupu k jiným počítačům v síti, ale jeho použití může být různorodé. Protokol SSL/TLS je používán například bankami pro posílání kritických informací.

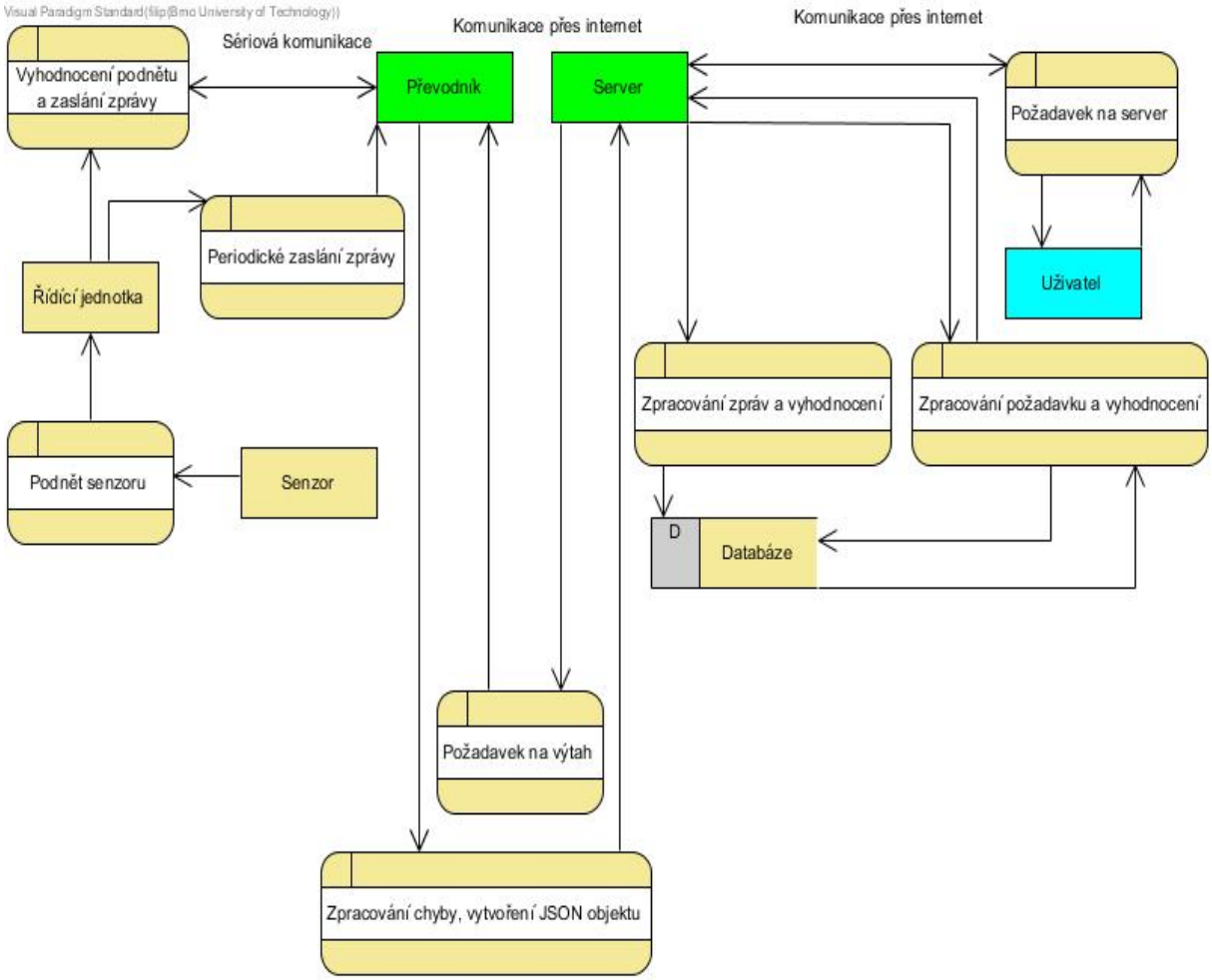
Z hlediska udržovatelnosti zabezpečení systému bylo zvoleno využití HTTPS či VPN. Vytvořený tunel u SSH protokolu se musí vždy "svázat"na konkrétní porty. Pokud bychom v serverové části vytvořili další služby na jiných portech, bylo by nutné přenastavit všechny převodníky v provozu a vytvořit pro ně další tunely pro komunikaci se serverovou částí. To není u VPN ani HTTPS potřeba. VPN technologie navíc nabízí nepřeberné množství konfigurace, což je pro budoucí udržování systému důležité.

Pro implementaci zabezpečení komunikace mezi serverem a klientem v této diplomové práci je nejvhodnější zvolit zabezpečené REST API (HTTPS). Komunikace s použitím této technologie poskytuje potřebné prvky zabezpečení a není náročná na implementaci. REST API komunikaci lze použít pouze v případech, kdy klient vždy navazuje spojení a server pouze odpovídá. Pokud se podíváme v podkapitole 3.2.1 na komunikační protokol řídicí jednotky, vidíme, že v tomto případě pro komunikaci zabezpečené REST API použít lze.

Pokud by byla komunikace mezi převodníkem a serverem implementována jiným způsobem, je možné zvolit jiný typ zabezpečení. Například TCP server neposkytuje žádné prvky zabezpečení. Pro jeho zabezpečení lze zvolit VPN technologii, kterou lze udržovat jednodušeji, než zabezpečení pomocí SSH protokolu. Implementací VPN je celá řada. Pro potřeby diplomové práce může být zvolena open source distribuce **OpenVPN**. Ta poskytuje dostačující zabezpečení a funkčnost ve srovnání s ostatními VPN distribucemi [6]. Protokol je vysoce konfigurovatelný a jeho OpenSSL knihovna podporuje celou řadu kryptografických algoritmů pro šifrování. Pro implementaci zabezpečení je sice nutné stáhnutí VPN klienta, ten je však kompatibilní s většinou systému včetně Raspbianu.

4.3 Architektura systému

V této části je možné vidět jednotlivé komponenty systému a způsob, jakým spolu komunikují. Příklad komunikace je zachycen na diagramu níže.



Obrázek 4.1: Diagram architektury systému

V obdélníku s hranami se vyskytují veškeré objekty systému, mezi kterými komunikace probíhá a jsou potřebné pro vyvíjený systém. Jedná se o senzory, řídicí jednotku, převodník a server. Dále se v systému nachází uživatel, který generuje požadavky na serverovou část. Části označené zeleně budou implementované včetně databáze v rámci diplomové práce.

Senzory, kterými výtah disponuje, mohou vyvolat podnět, který je řídicí jednotkou vyhodnocen jako chyba. Při výskytu chyby a jejím zpracováním řídicí jednotkou se zašle sériovou komunikací informace převodníku. Řídicí jednotka po vyhodnocení chyby provede určitou akci (například zablokuje výtah). Jednotka může zprávu zaslat i v jiných případech, což je uvedeno v popisu protokolu. Zprávu může řídicí jednotka posílat například periodicky pro otestování komunikace.

Převodník slouží jako prostředník mezi řídicí jednotkou a serverem. Zpracovává přijaté informace, převádí je do vhodného formátu (v této diplomové práci JSONu) a zasílá na stranu serveru. Zprávy, které nejsou zasílány serverové části, jsou ukládány do paměti

převodníku. Podobným způsobem probíhá komunikace ze serveru k řídicí jednotce. K převodníku mohou být v budoucnu připojeny další komponenty či moduly, kterými lze rozšířit systém.

Server zodpovídá za vyhodnocení veškerých zpráv řídicí jednotky a dotazů klientů. Pro persistenci dat spolupracuje s databází. Do databáze se neukládají veškeré zprávy. V diplomové práci jsou do databáze ukládány pouze zprávy chybové. Uživatel pracuje s webovým rozhraním, pomocí kterého zadává požadavky na server. Server zodpovídá za veškerou logiku aplikace. Provádí autentizaci uživatelů, kontroluje oprávnění uživatelů, spravuje systém notifikací apod.

4.4 Architektura serveru

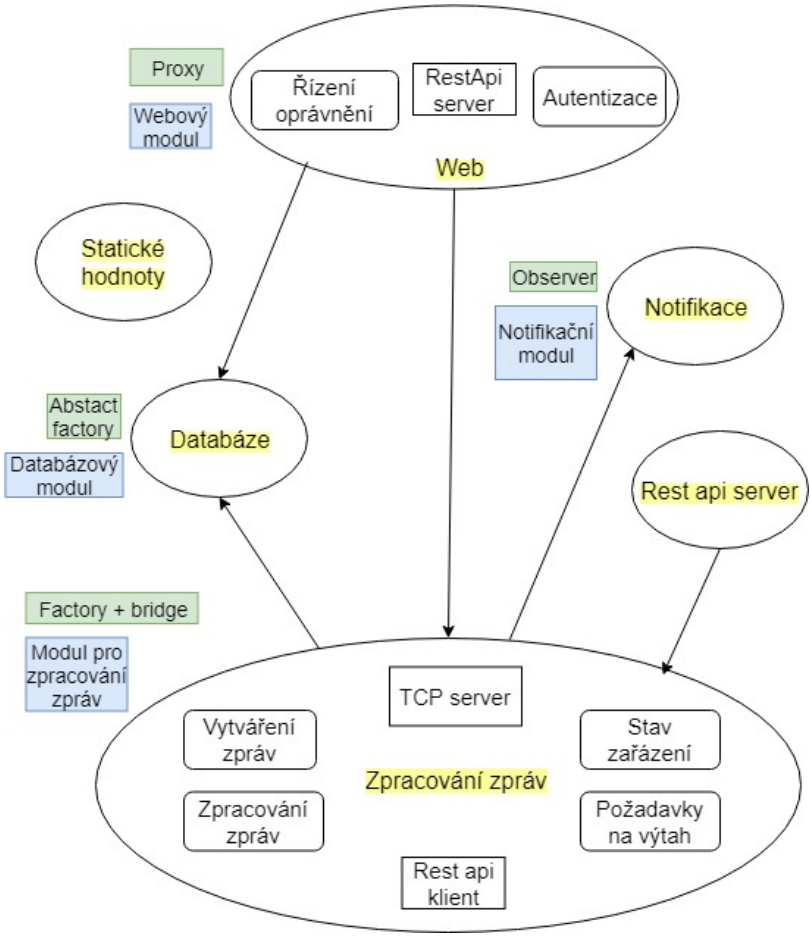
V podkapitole architektura serveru je znázorněn základní návrh systému, podle kterého bude probíhat implementace. První část je zaměřena na návrh struktury systému, druhá na návrh databáze.

4.4.1 Návrh implementace serverové části

Na diagramu níže lze vidět navrženou strukturu systému. Ovály označují moduly či balíčky, které budou v systému implementovány. Jejich názvy lze vidět v modrém obdélníku. Webový modul slouží pro zpracování všech požadavků uživatele. Veškeré požadavky klienta probíhají pomocí REST API architektury. Na diagramu lze vidět dva REST API servery. REST API server, který je součástí webového modulu, slouží pro klientské požadavky. Oddělený REST API modul bude určen pro komunikaci a obstarávání požadavků převodníku. Proto má tento modul závislost na modulu pro zpracování zpráv. Webový modul má tuto závislost z důvodu vzdáleného nastavení požadavků. Modul pro zpracování zpráv bude zajišťovat vyhodnocení některých požadavků uživatele a všech požadavků převodníku. REST API klient slouží pro komunikaci opačným směrem, tedy ze serveru k převodníku. Pojem REST API je často zmiňován i v kapitolách o implementaci převodníku a serveru. Proto je tento pojem pro lepší pochopení technologie stručně uveden v sekci 5.1. Protože je nutné pro zaslání REST API dotazu klientem znát adresu zařízení, bude převodník při inicializaci zasílat potřebné informace pomocí TCP serveru. Pokud by se komunikace přerušila, například v důsledku výpadku serveru, převodník na výpadek zareaguje. V takovém případě se bude pokoušet o znovuobnovení komunikace a opětovné zaslání informací. Pro zajištění persistence systému slouží modul databáze. Pro větší granularitu a nezávislost komponent je implementace databáze a notifikace oddělena do zvláštních modulů. Modul databáze je využíván webovým modulem a modulem pro zpracování zpráv. Modul pro zpracování zpráv databázi využívá pro ukládání chybových zpráv, neboť je nutné, aby bylo možné nahlédnout na chyby výtahu historicky. Modul notifikace bude zodpovídat, jak už název napovídá, za odeslání uživateli nastavených notifikací. Modul statických hodnot je viditelný pro všechny třídy a uchovává důležité statické informace, jako jsou IP adresy přiřazené serveru a podobně.

Jakým způsobem lze některé komponenty v systému implementovat, je v diagramu zvýrazněno zeleně. Modul databáze lze implementovat pomocí návrhové vzoru abstraktní továrny. Ten je vhodný pro systémy s menším počtem objektu v databázi. Výhodou tohoto vzoru je, že přechod na jiný typ databáze nezpůsobí velký zásah do systému. Řízení oprávnění může být implementováno pomocí návrhového vzoru proxy, který zaobaluje instance jiných objektů a řídí přístup k jejich rozhraní. Modul notifikace lze implementovat

pomocí návrhového vzoru pozorovatel. Zde si třída, která bude notifikace spravovat, udržuje na abstraktní úrovni kolekci svých pozorovatelů (observerů), které může upozornit. Pozorovatele budou v tomto případě zastupovat nastavené notifikace. Upozornění si lze představit jako zaslání notifikace uživateli. Pro zpracování zpráv lze využít návrhové vzory továrna a "bridge".



Obrázek 4.2: Diagram architektury serverové části

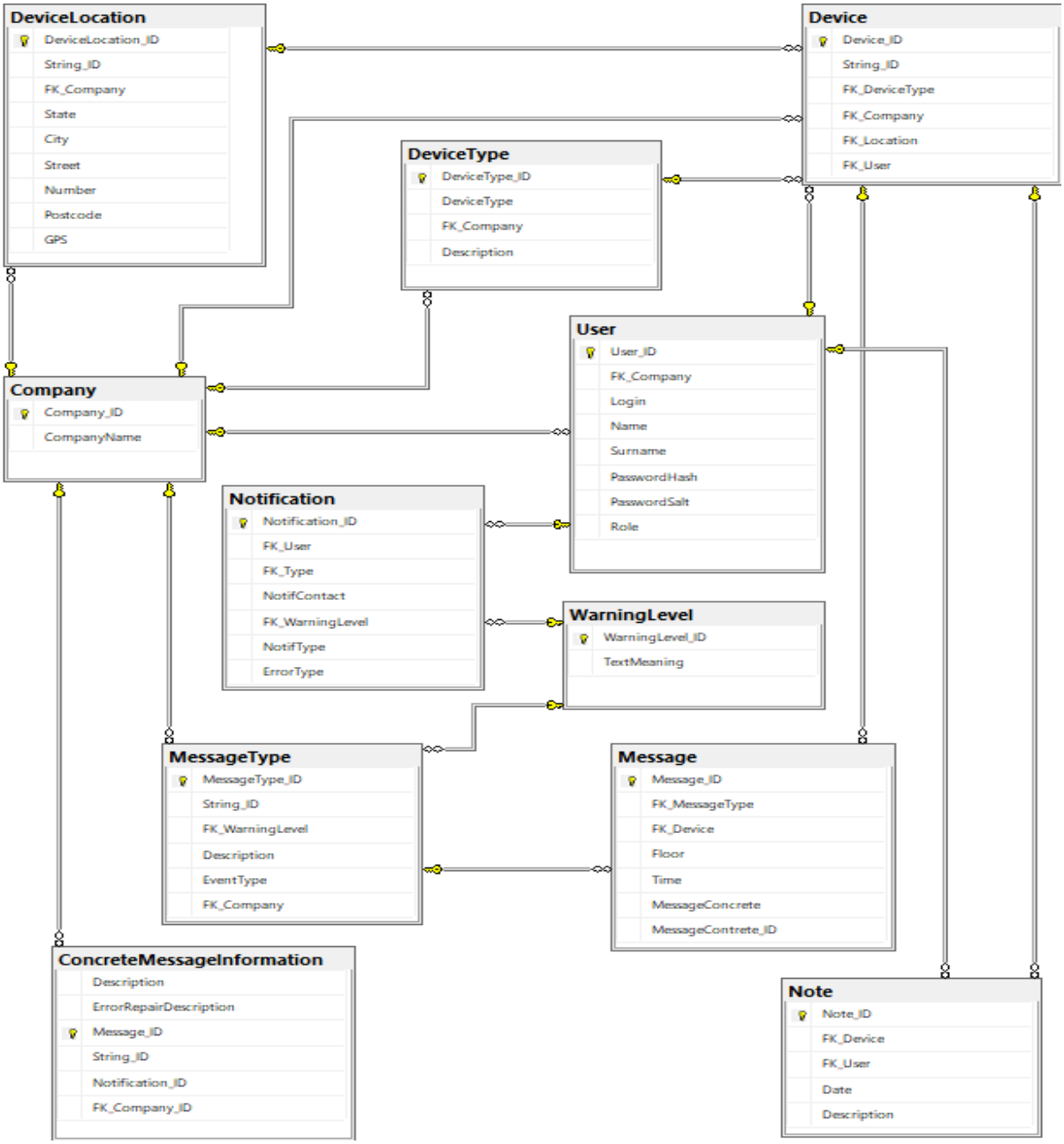
4.4.2 ER diagram

ER diagram zachycuje návrh databáze. Jsou zde vidět entity, pro které je důležité jejich uchování v čase a vztahy mezi nimi. Diagram je možné vidět na obrázku 4.3.
Do tabulky uživatelé jsou ukládáni všichni uživatelé, kteří se mohou autentizovat do systému. Pro autorizaci se u nich uchovává atribut s rolí. Dále jsou uchovány běžné informace jako je jméno, příjmení a login. Pro větší bezpečnost se hesla uživatele neuchovávají v textovém stavu. Hesla jsou šifrována a je ukládán jejich hašovací kód a takzvaná "sůl hesla"[1]. S kombinací obou záznamů lze dopočítat skutečné heslo uživatele. Databáze je připravena

[1] Vysvětlení kryptografických pojmů k dispozici na: https://gooroo.io/GoorooTHINK/Article/13023/The-difference-between-encryption-hashing-and-salting/2085#.Wu84GYiFPIU

pro přidávání poznámek uživatele k určitému zařízení. Pro poznámku je vytvořena entita s atributem datumu, který slouží pro zaznamenání časového údaje vytvoření poznámky a atributu poznámky, který bude uchovávat text poznámky. Uživateli může být přiřazena firma, pro kterou bude v databázi existovat oddělená entita. Firma bude přidělena pouze uživatelům s oprávněním "Pracovník". Pouze pracovník si bude moci nastavovat v systému notifikace. Konkrétnější informace o systému notifikací budou popsány v sekci implementace. Pro rozeznání typu notifikace je v entitě databáze příslušný atribut. Další atribut slouží pro uchování kontaktu, na který se budou informace zasílat. Kromě role pracovníka bude v systému existovat role správce a domovníka. Pouze domovníkovi bude možno přiřadit zařízení. Pracovníkům firmy se budou zobrazovat všechny zařízení, které jsou přiřazeny jeho firmě. Domovník uvidí v systému zařízení, které mu budou přiděleny. Entita zařízení je nejdůležitější částí databáze. Jak už název vypovídá, bude sloužit pro uchování všech informací o zařízeních, které bude systém vzdálené správy výtahu zákazníkům zobrazovat. Každé zařízení je rozeznáno pomocí identifikátoru. Identifikátor svým zařízením přiděluje firma Výťahy Zeva a je integrován v řídicí jednotce. Je zapotřebí počítat s tím, že v systému bude velký počet zařízení. Jak je vidět z části popisu protokolu, identifikátor je textový řetězec v hexadecimálním formátu. Indexování textových formátu je značně neefektivní. Proto se před uložením identifikátor překládá do číselného formátu, který je uložen a použit jako primární klíč zařízení. Zařízení patří určité firmě a může patřit také uživateli z důvodu popsaných výše. Ke každému zařízení je uchovávána informace o jeho poloze, k čemuž slouží tabulka s lokací zařízení. K lokacím bude možné ukládat běžné informace, jako je stát, město, ulice, číslo a číslo popisné. Protože lze počítat s tím, že výtahy jiných firem se mohou nacházet na stejné lokaci, jsou lokace vytvářeny vždy pro konkrétní firmu. Toho je využito při filtrování zařízení, kde bude možné zobrazení lokací dané firmy v systému. Po výběru konkrétní lokace se v systému zobrazí pouze zařízení, která se zde nacházejí. Zařízení jsou určitého typu (například "Výtah"). Typy zařízení jsou řešeny velmi podobným způsobem jako lokace zařízení. Jsou vytvářeny pro konkrétní firmu. V systému bude možné vytvořit filtr, který vyobrazí pouze zařízení určitého typu. Kromě poznámek je poslední složkou, která je přiřazena entitě zařízení, entita zprávy. Jedná se o zprávy, které jsou zasílány převodníkem. V tomto systému budou uchovávány pouze informace o chybových zprávách, ale databáze je připravena na to, aby bylo možné v případě požadavků firmy ukládat i jiné typy zpráv. V sekci implementace bude vysvětleno, proč se uchovávají pouze chybové zprávy, a jak se zpracovávají zprávy ostatní. Po přijetí zprávy se bude uchovávat informace o patře a časové razítko, které bude informovat o času přijetí zprávy na serveru. Dále je uchován obecný typ zprávy a konkrétní typ zprávy. Pro obecný i konkrétní typ zprávy byly vytvořeny entity, které uchovávají další informace přiřazené firmou. Obecný typ zprávy může být například již výše zmíněná chyba. Pokud by si firma přála ukládat i další zprávy, může o nich doplnit do systému informace, které se budou následně zobrazovat na webu. Každý typ zprávy má určitou úroveň závažnosti. Závažnost je uložena v separované entitě, ke které je kromě úrovně doplněn také popis. U obecného typu zpráv lze předpokládat, že se hodnoty nebudou obměňovat tak rychle, jako u konkrétních zpráv. Je nutné počítat s tím, že nové konkrétní zprávy budou vznikat rychlejším tempem. Entita, která uchovává konkrétní informace o dané zprávě, proto není propojena v databázi klíčem s entitou zprávy. Pokud by cizí klíč zprávy odkazoval na primární klíč konkrétní zprávy, nebylo by možné do systému přijatou zprávu uložit v případě, že pro ni neexistuje záznam. S tímto způsobem samozřejmě existuje riziko, že se do systému uloží zpráva s neplatným identifikátorem. Po přijetí zpráv je ovšem kontrolováno, zda při přenosu nedošlo k poškození zprávy, a proto je uložení zprávy s neplatným identifikátorem velmi nepravdě-

podobné. Při představování projektu firmě mohou být probrány obě varianty. Realizována bude varianta vyhovující firmě.



Obrázek 4.3: Diagram návrhu databáze serveru

4.5 Modularita systému

Raspberry pi a další dostupné minipočítače mají dostatečný výkon, a je proto možné kromě výtahů monitorovat větší počet zařízení. Průmyslová zařízení, včetně řídicí jednotky výtahu, často komunikují pomocí sériového rozhraní. Raspberry pi obsahuje několik sériových rozhraní. To ale nestačí pro velký počet zařízení, a proto je nutné použít řešení dostupná na trhu. V dnešní době existují sériové přepínače. K sériovému přepínači je možné připojit jedno hlavní zařízení a různý počet zařízení, které s ním budou komunikovat. Řešení pomocí sériového přepínače je kvůli vlastnostem sériové komunikace nepoužitelné, neboť současně mohou komunikovat pouze dvě zařízení. Na sériovém přepínači můžeme pouze staticky nastavit, které z připojených zařízení bude komunikovat s hlavním.

Problém s propojením více zařízení můžeme vyřešit pomocí převodníků, konkrétně převodníků převádějících sériovou komunikaci na ethernetovou. Ty jsou na trhu dostupné v různých provedeních. Existují verze, které mohou mít pouze jeden sériový port nebo verze s více sériovými porty. K převodníku můžeme připojit i více než 25 zařízení [8]. Pro bezdrátové zasílání dat z připojených zařízení existují WiFi verze převodníků. V případě, že bude v monitorované budově velký počet zařízení, není nutné mít ke každému zařízení minipočítač. Monitorované prvky v budově lze propojit s přijatelnější cenou. Data, která nemusí být zabezpečena či dále vyhodnocována, lze posílat přímo na server. Taková zařízení by byla pomocí převodníku, který převádí sériovou komunikaci, připojena k routeru. Zařízení, jejichž data je nutné zpracovávat, by musela být zaslána na minipočítač k vyhodnocení. Pro sériovou komunikaci lze v Raspberry pi využít všechny USB porty. Pokud by stále počet portů nestačil, situace by mohla být řešena například pomocí zmíněného převodníku sériové komunikace na ethernetovou. Jednotlivé převodníky sériové komunikace by bylo možné dále připojit ke klasickým routerům a v případě potřeby tímto způsobem vytvořit teoreticky libovolně velkou síť.

Příkladem komponentu, který můžeme zapojit s použití sériového převodníku do systému a rozšířit tak jeho funkčnost, může být RFID čtečka. RFID technologie je v tomto textu popsána v podkapitole 2.3.1. Raspberry pi obsahuje knihovny pro práci s RFID zařízeními. Pokud by výsledné řešení obsahovalo RFID čtečku pouze v kabině výtahu, je nejjednodušší připojit čtečku na sériový vstup Raspberry Pi. Raspberry pi může obsahovat jednoduchou databázi s přístupovými oprávněními a pomocí ní vyhodnocovat požadavky. RFID čtečku můžeme mít pro znemožnění přivolání výtahu neautorizovanými osobami umístěnou v každém patře. V takové případě lze podle počtu pater budovy zvolit převodník s vhodným počtem sériových rozhraní.

Kapitola 5

Implementace

Pro lepší pochopení implementace a funkčnosti systému budou nejdříve uvedeny komponenty na straně výtahu. To znamená testovací zařízení zastupující senzory a převodník. Až poté bude vysvětlena implementace serverové části. Jednotlivé komponenty je pro představu možné vidět v diagramu 4.1.

5.1 REST API architektura

REST (Representational State Transfer) je architektura rozhraní pro distribuované prostředí orientované na data. Umožňuje přistupovat k datům a provádět nad nimi CRUD operace[1] [7]. Je bezstavová a umožňuje paralelní zpracování obsahu. Pro REST je možné použít jakoukoli metodu přenosu, často je využito HTTP protokolu, který je použit i v této diplomové práci. K zavolání REST požadavku je nutné znát cestu ke zdroji (datům). Pro HTTP dotaz to je IP adresa spolu s dalším identifikátorem zdroje. Celá adresa pro zaslání požadavku může vypadat například takto: http://192.168.0.5/getParameter. IP adresu lze zaměnit za doménu, jak je to u protokolu HTTP běžné. Pro zaslání požadavků je dále nutné zvolit vhodnou metodu. Nejznámější metody, které protokol HTTP definuje, jsou GET, POST, PUT, DELETE. V diplomové práci jsou použity pro implementaci převodníku pouze metody GET - pro zaslání parametrů a POST - pro nastavení parametrů. Pro implementaci webového modulu jsou kromě GET a POST metod využity také metody PUT a DELETE. Metoda PUT slouží pro aktualizaci zdroje, metoda DELETE jak už název vypovídá pro jeho odstranění. REST je pouze rozhraní, které definuje, jakým způsobem by měla komunikace probíhat. Uživatelům používajícím REST nic nebrání v tom, aby v implementaci použili například pro aktualizaci zdroje metodu POST. Rozhraní by mělo být pro přehlednost a standardizaci dodržováno. REST nedefinuje, v jakém formátu mohou být data zasílána. Nejrozšířenějšími formáty pro přenos dat jsou jsou XML a JSON. Je možné využít další formáty, jako je například například YAML. Úspěšnost dotazu lze zjistit pomocí stavového kódu HTTP protokolu[2]. Kód 200 je vrácen v případě úspěšného požadavku, kód 201 při vytvoření nového obsahu. Další známý stavový kód je například 400, který je vrácen pokud server nemůže zpracovat požadavek. Stavový kód 401 informuje, že klient přistupuje k datům, ke kterým nemá přístup.

[1] Operace create, read, update, delete

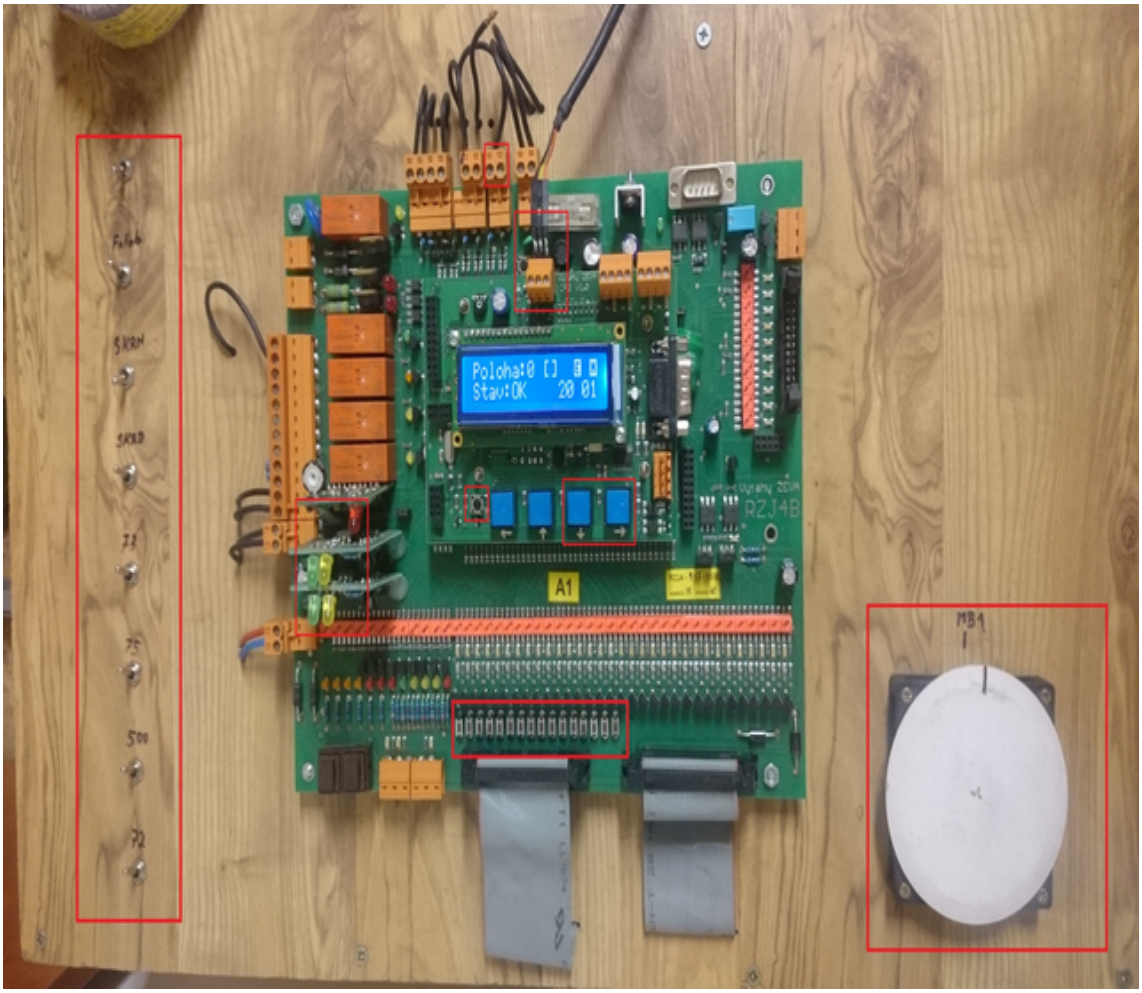
[2] Popis HTTP kódů k dispozici na: http://www.restapitutorial.com/httpstatuscodes.html

5.2 Testovací zařízení

Protože by bylo velmi obtížné provádět implementaci za reálného provozu výtahového systému, firma Výťahy Zeva zapůjčila k tomuto účelu testovací zařízení. Pomocí něho lze simulovat pohyb výtahu a některé z jeho chyb. Zapůjčené zařízení můžeme na zmíněném diagramu 4.1 považovat za řídicí jednotku a senzory.

Testovací zařízení je možné vidět na obrázku 5.1. V jeho středu se nachází displej s modrým podsvícením, na kterém jsou základní informace o stavu výtahu. Jedná se o stav, polohu výtahu, čas, stav dveří apod. Na obrázku se konkrétně výtah nachází v podlaží 0, je v pořádku a dveře jsou zavřeny (znázorněno symbolem "[]"). Informace o zapnutém monitoringu je na displeji nahoře vpravo. V podkapitole 3.2.1 je zmíněno, že je nutné veškeré zprávy potvrzovat. Pokud k potvrzení nedojde do krátkého časového limitu, symbol informující o zapnutém monitoringu začne periodicky blikat. Pokud je řídicí jednotka připravena pro zápis parametrů, které byly vzdáleně nastaveny, je místo symbolu informujícího o zapnutém monitoringu dočasně symbol "Z". Pokud by se výtah nacházel v chybovém stavu, místo symbolu "OK"by bylo na displeji vyobrazeno "Ex", kde x označuje číslo chyby. Důležité části jsou na obrázku označeny červeným obdélníkem. Tlačítka pod displejem slouží k ovládání řídicí jednotky, a některé z nich mají další funkčnost. Proto, aby nebylo možné zneužít systém vzdáleného nastavení parametrů, se musí veškeré vzdálené nastavení potvrzovat technikem ručně na řídicí jednotce. Pro potvrzení je nutné podržet tlačítko pod displejem nejvíce vpravo. Potvrdit vzdáleně nastavené parametry lze pouze v případě, že je na displeji symbol "Z". Pro vypnutí či zapnutí monitoringu slouží tlačítko nalevo od tlačítka pro potvrzení. Pro resetování řídicí jednotky slouží označené černé tlačítko, nacházející se nalevo od tlačítek pro ovládání. Nad displejem je možné vidět místo, kde je řídicí jednotka propojena s převodníkem. Pomocí spínačů označených nalevo testovacího zařízení lze vyvolat a simulovat určité stavy výtahu. Poslední čtyři spínače simulují sepnutí bezpečnostních obvodů, další dva tlačítka simulují, že je výtah v nejspodnějším, respektive v nejvrchnějším podlaží. Další spínač simuluje překážku, která by se v reálném provozu nacházela mezi šachetními dveřmi. O sepnutí bezpečnostních obvodů informují označené diody napravo od spínačů. Chybu lze simulovat například tak, že se rozpojí některý z bezpečnostních obvodů a zvolí libovolné patro. Například při rozepnutí bezpečnostního obvodu 500 a navolení patra se na displeji řídicí jednotky objeví chyba E9, což svědčí o skutečnosti, že se nepodařilo uzavřít obvod 500. Při zapnutém monitoringu je odeslána chybová zpráva přes sériovou jednotku. K simulaci přivolání patra slouží tlačítka, nacházející se ve spodní části řídicí jednotky. Pohyb výtahu je simulován otáčením kolečka v pravé části testovacího zařízení.

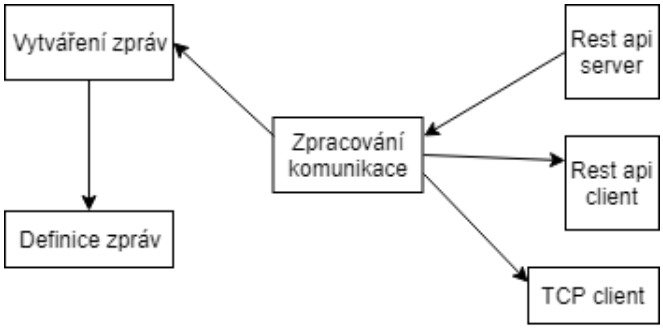
Testovací modul má nedostatky, na které je nutné upozornit. Není schopen simulovat veškeré chyby, které se u reálného výtahu vyskytují. Testovací modul také neumí simulovat otevření dveří kabiny při dojezdu do patra. To způsobí vyvolání chyby E14 - neotevření šachetních dveří do časového limitu. Firma Zeva již byla na tento nedostatek upozorněna a pracuje na jeho vyřešení. Pokud chceme simulovat otevření dveří po dojezdu do patra, je nutné ihned po příjezdu otevřít obvod 75.



Obrázek 5.1: Ukázka testovacího zařízení

5.3 Převodník

K funkci převodníku bylo vybráno zařízení Raspberry Pi, konkrétně model 3 B+. S testovacím zařízením je propojen pomocí TTL/USB kabelu. Stejný kabel bude použit případně pro připojení řídicí jednotky reálného výtahu s převodníkem. V rámci diplomové práce je připojen převodník k internetu pomocí WiFi. V reálném provozu se lze k internetu připojit podle požadavků klienta a vybavenosti strojovny. Ve strojovně se obvykle nacházejí přepínače, které přeposílají síťový provoz obyvatelům v dané budově. Tohoto faktu by se dalo využít k připojení převodníku k internetu. Pokud by se ve strojovně nenacházel žádný zdroj signálu, bude připojení k internetu řešeno zakoupením SIM karty a připojením k 3g či LTE síti. Na obrázku 5.2 jsou vidět základní komponenty architektury převodníku. Pro implementaci převodníku byl zvolen jazyk Python. Architektura systému je rozdělena na 6 hlavních částí.



Obrázek 5.2: Architektura převodníku

5.3.1 Zpracování dat

Hlavní částí, ve které je prováděná největší část logiky, je třída pro zpracování komunikace. Tato třída je implementována pomocí návrhového vzoru singleton[3] a je dostupná i pro ostatní třídy či moduly. Toho je využito pro vzdálené nastavení či zaslání parametrů výtahu. Implementace vzdáleného nastavení parametrů bude popsána dále.

Po vytvoření instance třídy pro zpracování komunikace, jsou periodicky čtena příchozí data z řídicí jednotky výtahu. Pokud jsou ve vyrovnávací paměti data, jsou okamžitě přečteny. Protože mají datové zprávy protokolu řídicí jednotky stejnou velikost, je možné číst konstantní velikost příchozích dat - 22 bajtů. Příchozí data nejsou na server přeposílána v surovém formátu, ale zpracovávají se v převodníku. Ihned po přijetí zprávy se jednotlivé bajty konvertují do ASCII řetězce, ve kterém je možné vidět jednotlivé hodnoty. K vytvoření konkrétní zprávy slouží třída k vytváření zpráv. Pro zaznamenání skutečnosti, že probíhá vzdálené nastavení či přijetí parametrů, ve třídě existují pravdivostní hodnoty. Definice tříd pro jednotlivé zprávy, které zasílá řídicí jednotka, se nacházejí v modulu definice zpráv. Na základě typu zprávy se vytváří instance, kterou systém převodníku dále zpracovává. Informace o jednotlivých zprávách jsou poskytnuty v kapitole 3.2.1. Vytvořená instance obsahuje atributy, ve kterých jsou uchovány jednotlivé hodnoty datové zprávy. Informace jsou pro lepší správu stále uloženy v ASCII řetězci. Pro zpětné získání zprávy z instance, implementují všechny třídy metodu abstraktní třídy k tomu procesu určenou.

Veškeré zprávy z řídicí jednotky výtahu je nutné potvrzovat. Ještě před zasláním potvrzovací zprávy jednotce je nutné vypočítat kontrolní XOR součet původní zprávy. Tím se ověří, jestli nebyla narušena integrita. Pokud je XOR zprávy správný, lze vytvořit konkrétní potvrzovací zprávu a zaslat ji řídicí jednotce. V opačném případě je zpráva zahozena a nepokračuje se v jejím zpracování. Pokud zpráva není potvrzena do časového limitu 30 sekund, řídicí jednotka posílá nepotvrzenou zprávu znova. Kontrolní součet je kontrolován u všech zpráv, které jsou zasílány na server.

Zpráva s počítadly není serverové části zaslána ihned, neboť není nutné její uložení do databáze a další vyhodnocení. Zpráva s počítadly je uložena do paměti převodníku a ihned potvrzena. Ke zpracování jsou na server zasílány zpráva o události a testovací zpráva. Zpráva o události informuje o vzniklé chybě a v testovací zprávě jsou jednak uloženy informace o stavu výtahu a také se touto zprávou testuje, že je komunikace mezi řídicí jednotkou a serverem v pořádku. K zaslání zpráv je využito REST rozhraní implementované ve třídě REST API klienta. Serveru jsou zasílány pomocí výše uvedené třídy vytvořené

[3]Popis návrhového vzoru singletion dostupný na: https://sourcemaking.com/design_patterns/singleton

instance zpráv ve formátu JSON. Pokud při zpracování v serverové části nedojde k chybě, postupuje se způsobem popsaným výše. Je vypočítán XOR zprávy a s jeho platnou hodnotou, je zaslána konkrétní potvrzovací zpráva řídicí jednotce. Bližší informace o tom, jak probíhá zpracování zpráv na straně serveru je uvedeno v kapitole 5.4.3.

5.3.2 Vzdálené nastavení a přijetí parametrů

Kvůli vlastnostem REST API byla architektura použita i na zasílání požadavků pro vzdálené nastavení či zaslání parametru. Pro tyto účely je v převodníku implementován REST API server, který přijímá požadavky serverové části systému. V tomto typu komunikace navazuje spojení vždy strana, která zasílá požadavek (zde server). Server ovšem nemá k dispozici informace potřebné k zaslání dotazu. K předání informací o převodníku slouží třída TCP klienta. REST server převodníku naslouchá na všech IP adresách přiřazených síťové kartě. Serverové části systému je poskytována pouze jedna IP adresa, na kterou je možné požadavek zaslat. K zavolání dotazu je nutné znát celou cestu k obslužné metodě, která požadavek zpracovává. V tomto případě je k IP adrese připojen řetězec "getParameter"k zaslání požadavků pro zaslání parametrů respektive "setParameter"pro vzdálené nastavení parametrů. Pro zaslání parametrů je využita HTTP metoda GET, pro nastavení metoda POST. Tímto způsobem jsou serverové části zasílány i cesty k ostatním metodám, které server výtahového systému využívá. Poslední informací, která je zaslána, je identifikátor převodníku, který je uveden v prvních třech bajtech každé zprávy. Z řádků výše je patrné, že se celý systém musí spouštět postupně. Veškeré inicializace jsou prováděny v samostatné třídě. Nejprve je vytvořena instance pro zpracování komunikace, které je předán požadavek na vrácení identifikátoru řídicí jednotky. V momentě, kdy je převodníku z řídicí jednotky zaslána první zpráva, je předán identifikátor výtahu. Tímto způsobem je získán identifikátor, který je spolu s dalšími informacemi zasílán serverové části. IP adresa serveru, na který se informace zasílají, je specifikována v kódu. V novém vlákně je spuštěn "pracovní"proces TCP klienta, kterému jsou získané informace předány. V něm jsou po vytvoření spojení se serverem zaslána informační data ve formátu JSON. Spojení mezi klientem a serverem se uchovává i po zaslání informací.[4] Pokud se spojení přeruší, například z důvodu výpadku serveru, klient se snaží spojení opět navázat a zaslat data. Zaslaná data jsou vyhodnocena v serverové části, která po zpracování obsahuje všechny potřebné informace k zaslání požadavku převodníku. Po vytvoření TCP klienta je v novém vlákně vytvořena instance pro zpracování požadavků serverové části.

K přijímání požadavků slouží třída, která implementuje REST API server. Metody, které zpracovávají požadavky, využívají instanci pro zpracování komunikace. Ta obsahuje metody k tomuto účelu určené. Komunikace probíhá následujícím způsobem. REST API třída příjme požadavek serveru a zavolá příslušnou metodu. Za veškeré vykonání logiky je od této chvíle zodpovědná třída pro zpracování komunikace. Ta po zpracování vrátí buď požadované data nebo chybu. Podle úspěšnosti předá REST API třída serveru data se stavovým HTTP kódem 200 nebo chybu se stavovým kódem 500. Požadavek pro zaslání či nastavení parametrů nemůže být vyřízen okamžitě. Je nutné, aby nejprve řídicí jednotka zaslala testovací zprávu. Na ni převodník odpovídá zprávou se žádostí o nastavení či zaslání parametrů. Celý proces komunikace převodníku s řídicí jednotkou lze vidět na diagramu 3.2. Požadavek tedy neproběhne okamžitě a lze předpokládat, že může probíhat několik požadavků paralelně. Pokud probíhá nastavení parametrů a je zaslán požadavek na další

[4] Spojení bylo vytvořeno jako TCP keep alive. Popis tohoto druhu spojení je dostupný na: http://tldp.org/HOWTO/TCP-Keepalive-HOWTO/overview.html

nastavení či zaslání parametrů, je toto chování vyhodnoceno jako chybné. V prvním případě hrozí, že se ihned přepíšou právě nastavené parametry. Ve druhém případě hrozí, že by klientovi byly zaslány neaktuální data. Pokud naopak probíhá požadavek na zaslání parametrů, chyba nastane jen v případě, že je paralelně zaslán požadavek na nastavení parametrů. Více požadavků na zaslání parametrů je zpracováno. Zpracování je realizováno způsobem, který je velmi podobný řešení problematiky konzumenta a producenta v paralelním programování[5]. Všechna vlákna, která zažádala o zaslání parametrů jsou uspána a čekají na odbavení. Instance pro zpracování komunikace vyřizuje požadavek, čte data z vyrovnávací paměti pro sériovou komunikaci a reaguje na příchozí zprávy způsobem nastíněným výše. Řídicí jednotka posílá dvě zprávy s parametry. První zpráva obsahuje parametry, které je možné vidět v příloze A, na hodnoty druhé zprávy lze nahlédnout v poslední části přílohy. Z obou zpráv jsou vytvořeny instance, které jsou přiřazeny atributům třídy, která výsledek zaobaluje a která je zaslána serveru. V momentě, kdy jsou dostupná potřebná data, jsou vlákna probuzena. Po probuzení vlákna přečtou instanci s parametry, a výsledek je zaslán REST API serverem převodníku. Tak jako všechny ostatní zprávy, jsou data s parametry zaslána ve formátu JSON. Velmi podobným, ale opačným způsobem probíhá také nastavení parametrů. Převodníku jsou zaslána serializovaná data v JSON formátu. Po deserializaci mají data podobu instance třídy se dvěma atributy. Atributy v tomto případě obsahují zprávy v bajt formátu, které je možné ihned zaslat pro zpracování řídicí jednotce. Protože při nastavení parametrů nelze z důvodu uvedených výše paralelizovat, není nutné řešit současný přístup vláken. Po nastavení parametrů je zpět na server zaslána HTTP správa s kódem, který reflektuje průběh operace. Po vyřízení požadavků instance pro zpracování komunikace opět pracuje běžným způsobem.

5.4 Server

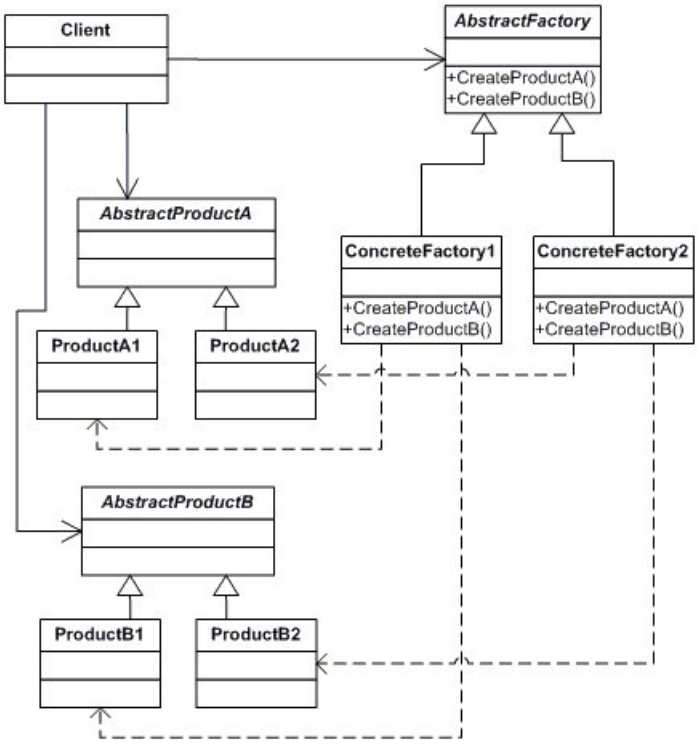
V této kapitole bude detailněji popsána implementace serverové části. Popis bude navazovat na návrh serverové architektury v kapitole 4.4. Backend je implementován v jazyce C#, pro persistenci dat byla zvolena relační databáze MS-SQL. Frontend je implementován s použitím frameworku Angular 4 a jazyků Typescript, HTML a CSS. Pro jednodušší stylování webových stránek a pro optimalizaci velikosti stránek pro různá zařízení, je použit framework Boostrap. Pro přehlednost budou komplexnější moduly popsané v návrhu uvedeny v oddělených odstavcích. Moduly budou postupně představovány tak, aby se v popisu implementace objevovaly pouze již vysvětlené části. V úvodní části některých modulů bude stručně objasněn návrhový vzor, který byl použit pro implementaci.

5.4.1 Modul databáze

K implementaci databázového modulu byl zvolen návrhový vzor abstraktní továrna. Pomocí tohoto vzoru lze flexibilně zvolit typ, který budou produkovat abstraktní továrny. V této diplomové práci je využívána flexibilita k volbě typu databázového serveru. Na obrázku níže lze vidět UML návrh abstraktní továrny. Abstraktní továrna obsahuje abstraktní třídu, která definuje metody, které musejí implementovat konkrétní továrny. Abstraktní továrna zodpovídá za vytvoření konkrétních továren, které zapouzdřují definovanou funkcionalitu. Tyto třídy mají sadu společných vlastností, které je předurčují ke společnému použití. Pokud bychom chtěli změnit implementaci systému, stačí pouze zaměnit instanci,

[5] Popis problému producenta a konzumenta dostupný na: https://www.agiliq.com/blog/2013/10/producer-consumer-problem-in-python/

kterou vytváří abstraktní továrna. Tímto způsobem se zachová společná sada vlastností, které generují konkrétní továrny, ale změní se jejich implementace. Konkrétní továrny vytvářejí produkty definované abstraktní třídou. Produkty implementují rozhraní, ke kterým v aplikaci přistupujeme. V diagramu přistupuje k rozhraním třída klient. V následujícím textu bude vysvětleno použití návrhového vzoru abstraktní továrny při implementaci databázového modulu.



Obrázek 5.3: UML diagram návrhového vzoru abstraktní továrna. Převzato z: [1].

Abstraktní továrna definuje entity, od kterých je v diplomové práci požadována persistence. V diplomové práci jsou to například uživatelé, notifikace, zařízení apod. Za vytvoření těchto entit jsou zodpovědné konkrétní továrny. Kromě konkrétních továren se vytváří také instance zodpovědná za vytvoření a poskytnutí připojení k databázi. To je potřebné pro vykonávání databázových operací. Typ instance konkrétní továrny je vytvořen podle nastavení systému. Typ databáze je specifikován v souboru app.config. V něm je uveden také connection string[6], který je nezbytný pro vytvoření připojení k databázi. Pro tuto diplomovou práci byla zvolena databáze MS-SQL, která je dobře podporovaná pro jazyk C#, neboť jsou tyto produkty od společného poskytovatele firmy Microsoft. Instance, které vytváří konkrétní továrny, implementují operace sloužící pro komunikaci s relační databází. Velice často jsou to CRUD operace.

Kromě abstraktní továrny byl pro implementaci databázového modulu použit návrhový vzor singleton. Při inicializaci systému je vytvořena instance konkrétní továrny a instance poskytující připojení k databázi podle typu specifikovaném v souboru app.config. Instance jsou udržovány v oddělených singletonech, přes které je k nim možné v systému přistupovat. Při specifikaci používané v této diplomové práci první singleton obsahuje in-

[6] https://docs.microsoft.com/cs-cz/dotnet/framework/data/adonet/connection-string-builders

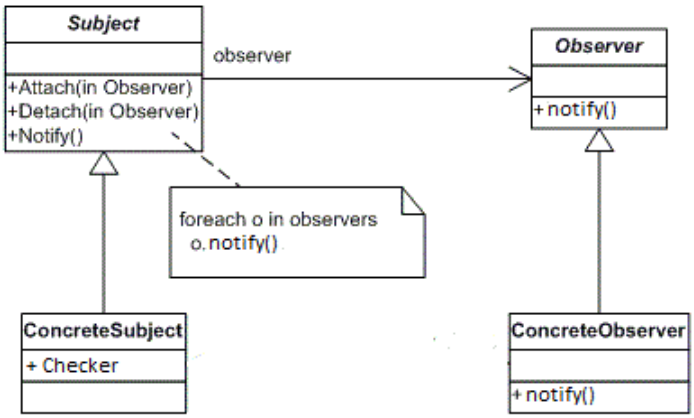
stanci konkrétní MS-SQL továrny. Ta vytváří instance implementující databázové operace na základě definovaného rozhraní (např. MSSQLUživatelDAO implementující rozhraní IuživatelDAO). Vytvářené instance používají k databázovým operacím druhý typ singletonu, který poskytuje připojení k MS-SQL databázi. Připojení je použito při každém přístupu k databázovému serveru.

Pro vytvoření všech entit v databázovém serveru byl použit nástroj Microsoft SQL Management Studio. Ten umožňuje v přehledném uživatelském rozhraní vytvářet tabulky a jednotlivé vztahy. Studio poskytuje možnost testování SQL dotazů. Proto byly před implementací vzdálené diagnostiky dotazy vždy zkoušeny v tomto programu. Další užitečnou vlastností Management Studia je možnost vygenerovat z navržené databáze ER diagram. Generovaný diagram je možné vidět v kapitole 4.4.2.

Při jakékoliv operaci s databází se v systému přistupuje pouze k definovaným rozhraním (např. IuživatelDAO). Jak je možné vidět na příkladech uživatele, jsou dodržovány určité konvence k udržení pořádku v kódu. Konkrétní továrny začínají vždy názvem poskytovatele databáze. Ve všech třídách či rozhraních, které implementují logiku databázového modulu, se objevuje označení DAO. Tyto konvence jsou dodržovány kvůli přehlednosti a odlišení tříd implementujících logiku databáze od tzv. kontejnerů. Na diagramu s návrhovým vzorem přistupuje k rozhraním definujícím databázové operace třída klient. V modulu databáze přistupuje k těmto rozhraním vždy třída - kontejner, která uchovává informace o databázových entitách v systému vzdálené diagnostiky. Může se jednat například o kontejner uživatele. Tyto třídy obsahují atributy, které se kryjí s atributy entit v databázi. Kromě nich obsahují další atributy, které se mohou například dopočítat z databázových dat. Pro přístup k databázi implementují kontejnery metody, které přistupují k rozhraním definujícím databázové operace. Kromě metody pro uložení, kde musí existovat před uložením do databáze instance s informacemi, jsou veškeré metody kontejnerů statické. Data v kontejneru "uživatel"s požadovanými hodnotami, lze uložit zavoláním metody ulož(). Implementace metody ulož vypadá v kódu následovně: `"IDAOFactory.GetFactory().GetUserDAO().save(this)"`. IDAOFactory je singleton, který v metodě GetFactory() vrací konkrétní instanci továrny. Konkrétní továrna obsahuje metody, které vracejí instance entit implementující databázové operace (např. MSSQLUživatelDAO). Metoda ulož volaná na tyto instance implementuje databázový dotaz a interně pracuje se singletonem pro databázové spojení. Stejným způsobem jsou pro přístup k databázovému serveru implementovány i ostatní metody všech kontejnerů v systému.

5.4.2 Modul notifikace

Implementace modulu notifikace byla inspirována návrhovým vzorem pozorovatel. Tento návrhový vzor umožňuje udržovat kolekci objektů, které při změně stavu reagují voláním svých interních metod. V diagramu níže můžeme vidět abstraktní třídu předmět, která definuje metody pro přidání, odebrání a upozornění pozorovatelů. Konkrétní předmět, který implementuje všechny výše zmíněné metody, udržuje kolekci všech pozorovatelů. Zavolání metody pro upozornění způsobí, že je zavolána stejně pojmenovaná metoda na všechny záznamy v kolekci. V kolekci jsou uloženy objekty, které implementují rozhraní, definující metodu pro upozornění. Tím je zajištěna větší abstrakce spravovaných objektů, neboť konkrétní pozorovatelé mohou mít metodu pro upozornění implementovanou odlišně.



Obrázek 5.4: UML diagram návrhového vzoru pozorovatel. Převzato z: [4].

V systému vzdálené diagnostiky se nachází abstraktní třída pro správu notifikací, která definuje metody popsané výše: přidej, odeber, upozorni. Metody přidej a odeber slouží pro přidávání a mazání notifikací z kolekce konkrétní instance. V tomto systému slouží pro tyto účely třída pro správu notifikací firmy Zeva. Pokud bychom do systému přidali další firmu, bylo by možné přidat do systému další konkrétní instance pro správu notifikací. V systému vzdálené diagnostiky existují dva druhy konkrétních notifikací. Prvním typem jsou emailové notifikace, druhým typem jsou SMS notifikace. Pokud je potřeba upozornit pracovníky firmy Zeva na vzniklý problém, v instanci pro správu Zeva notifikací je na abstraktní úrovni zavolána metoda notifikuj u všech objektů v kolekci. Konkrétní notifikace v kolekci (SMS či emailové notifikace) implementují své metody a reagují na zavolání odlišně.

Konkrétní instance pro správu notifikací si kromě kolekce všech notifikací udržují v atributu instanci sloužící pro zpracování a vyhodnocení notifikací. Ta je inicializována při vzniku konkrétní instance pro správu notifikací a je předána každému pozorovateli při upozornění. Funkčnost bude detailněji popsána při popisu implementace metody upozornění u konkrétních notifikací. Metoda upozornění v instanci pro správu notifikací přijímá jediný parametr s informacemi o přijaté zprávě, na kterou je nutné upozornit. Jakým způsobem probíhá vyhodnocení přijaté zprávy a kdy probíhá zavolání upozornění na zprávu, bude popsáno v podkapitole 5.4.3. Pokud část pro zpracování zprávy vyhodnotí, že je nutné na zprávu upozornit, vytváří novou instanci s informacemi. Nová instance je předána při volání metody upozorni a obsahuje všechny potřebné informace k vyhodnocení notifikace. Typ zprávy a úroveň závažnosti je použit při vyhodnocení, zda se má notifikace poslat či ne. Aby byly notifikace co nejvíce informativní, obsahuje zpráva další informace, jako jsou například identifikátor a typ zařízení, všechny údaje o lokaci zařízení a typ poruchy.

Při spuštění systému probíhá inicializace modulu notifikace a je vytvořena instance pro správu notifikací firmy Zeva. Z databáze jsou získány pomocí kontejneru databázového modulu všechny notifikace pracovníků firmy Zeva. Ty jsou předány instanci pro správu notifikací. Entity notifikace v databázi obsahují atribut, který definuje jejich typ. Kromě typu obsahují notifikace v databázi také závažnost či konkrétní typ zprávy. Z informací z databáze se podle typu notifikace vytvářejí konkrétní instance emailových či SMS notifikací. Instance obsahují atributy, ve kterých jsou uloženy všechny informace z databáze. Inicializace i zpracování emailových a SMS notifikací jsou dále odlišné. Nejprve bude popsán způsob inicializace emailových notifikací.

V C# jsou k dispozici knihovny, které lze využít k posílání emailů. Knihovny defi-

nují třídu SMTP klienta a SMTP zprávy. Při vytváření SMTP klienta je zapotřebí předat emailovou adresu, ze které se email posílá a heslo k emailovému účtu. Tyto informace jsou uvedeny ve statické třídě, ke které má modul notifikace přístup. Při vytváření SMTP zprávy je nutné předat emailovou adresu adresáta a odesílatele. Emailová adresa adresáta je získána z databázových informací.

C# neobsahuje knihovny pro odesílání SMS zpráv. Pro tyto účely však existuje velké množství platforem, které lze do systému integrovat. Některé platformy poskytují knihovny, které se integrují do zdrojových kódů. U velkého množství platforem stačí pro zaslání SMS zprávy odeslat přesně definovaný REST API dotaz. Tímto způsobem pracuje i SMS platforma Clickatel, která byla pro implementaci SMS zpráv použita. K využívání platformy je nutné se registrovat na jejich webových stránkách a vytvořit si zde účet. Clickatel umožňuje po vytvoření účtu zadat tři čísla, na kterých je možné testovat funkčnost, a které nejsou zpoplatněny. Pro účely diplomové práce je registrováno pouze číslo mobilního telefonu autora systému. Po zadání čísel je vytvořen identifikační kód, který je zasílán v obsahu REST API dotazu. Při zasílání SMS na více, než tři čísla je nutné přepnout účet do "netestovacího"režimu. V tom případě je účtována každá zaslaná SMS. Jedna SMS zpráva stojí méně, než jednu korunu. Při inicializaci SMS notifikací je vytvořen HTTP klient, kterému je při vytvoření předán identifikační kód popisovaný výše a Clickatel URI ve tvaru: "https://platform.clickatell.com/". Dále je vytvořen SMS objekt, ve kterém je definován příjemce. Takto nastavené objekty jsou připraveny pro odeslání notifikací.

V následující části bude popsáno, jak probíhá vyhodnocení metody upozornění u konkrétních notifikací. Začátek vyhodnocení je pro emailové i SMS notifikace stejný. V parametrech jsou předány dva atributy. Prvním atributem je zpráva s informacemi potřebnými pro zpracování notifikace, druhým atributem je instance sloužící pro zpracování notifikací. Instance pro zpracování je využívána, neboť vyhodnocení notifikací u různých firem může být odlišné. Obsahuje metodu vyhodnoť, která vrací pravdivostní hodnoty: pravdu, pokud se má notifikace zaslat, jinak nepravdu. Tato metoda přebírá u implementace firmy Zeva 3 parametry. Prvním je zpráva s informacemi, druhým úroveň závažnosti a třetím konkrétní zpráva, při které se má notifikace zaslat. Poslední dva parametry jsou definovány uživatelem, a jsou získány při inicializaci modulu. Vyhodnocení pro firmu Zeva probíhá následovně. Pokud je u notifikace definována závažnost, je srovnána se závažnosti zprávy. Pokud jsou závažnosti totožné, metoda vrací pravdu. Pokud nikoliv, je kontrolován konkrétní typ zprávy. Pokud je konkrétní typ nastaven a je stejný jako typ zprávy, je vrácena pravda. Pokud nenastane ani jedna z výše popsaných situací je vrácena nepravda. Instance pro zpracování zprávy obsahuje kromě metody pro zjištění, jestli se má notifikace zaslat, dvě metody. První metoda získává předmět pro emailovou zprávu a druhá tělo zprávy jak pro emailovou, tak pro SMS notifikaci. Další vyhodnocení upozornění pro emailové i SMS notifikace je velice podobné. Oba druhy používají k upozornění již inicializované objekty. U emailu i SMS notifikace je získáno pomocí instance pro zpracování notifikace tělo zprávy. U emailu je pomocí stejné instance získán předmět zprávy. U emailu je před-inicializované SMTP zprávě dodáno tělo a předmět. SMTP správa je předána asynchronní metodě SMTP klienta, která slouží k odeslání emailu. U SMS notifikace je při upozornění přidáno tělo zprávy k SMS objektu zprávy. SMS objekt je serializován do formátu JSON a předán do metody HTTP klienta, která zasílá REST dotaz. Tím je delegováno zaslání SMS zprávy na platformu Clickatel. Je nutné podotknout, že stav všech notifikací není měněn jen na základě informací z databáze při inicializaci. Pokud je na webu přidána uživatelem nová notifikace, je dynamicky přidána do kolekce stejnou metodou, jako jsou přidávány informace z databáze. To samé platí

i při vymazání notifikace uživatelem. Kromě vymazání z databáze probíhá také smazání notifikace v kolekci.

5.4.3 Modul pro zpracování zpráv

Spolu s modulem REST serveru implementuje modul pro zpracování zpráv veškerou logiku nutnou pro komunikaci s převodníkem a zpracování dat z převodníku. Modul se skládá z více částí, které budou popsány v následující části. Nejprve bude popsáno, jakým způsobem jsou zpracovány zprávy z převodníku. Poté implementace vzdáleného nastavení parametrů.

Pro přijímání zpráv z převodníku slouží modul REST API server. Ten pro svou velikost nebude popsán separátně, ale bude stručně popsán nyní. REST server má po vytvoření definovanou URL, na které naslouchá. Na stejnou adresu převodník zasílá veškeré zprávy. Pro všechny zprávy existuje pouze jedna URL adresa s metodou, ve které probíhá zpracování. Za veškeré zpracování zodpovídá modul pro zpracování zpráv. Modul REST serveru slouží pouze pro přijetí zprávy a zaslání odpovědi.

Po přijetí je nejprve vytvořena instance zprávy, která je v systému dále zpracována. Pro tento účel slouží třída, která je implementována pomocí návrhového vzoru abstraktní továrny. Z databáze jsou získány informace o konkrétní zprávě včetně firmy spravující zařízení, ze které zpráva pochází. Pomocí tohoto údaje se vytvoří konkrétní továrna firmy. Ta vytváří konkrétní typy zpráv dané firmy. Instance zpráv obsahují informace získané z databáze. Protože jsou řídicí jednotkou firmy Zeva na server zasílány pouze testovací a poruchové zprávy, instance Zeva továrny vytváří pouze dva druhy instancí. Vytvořené instance obsahují metodu, která umožňuje z jejich dat získat potvrzovací zprávu zasílanou převodníku. Po vytvoření jsou instance předány objektu třídy pro zpracování zpráv. Podle typu zprávy je vytvořena konkrétní instance pro zpracování. Instance zpráv firmy Zeva jsou v konstruktoru předány instanci pro zpracování firmy Zeva. V objektu, který zpracovává zprávy, je dále vytvořena instance, která zodpovídá za ukládání zpráv do databáze pro konkrétní firmu. Instance, ukládající zprávy pro firmu Zeva, obsahuje pouze metodu pro uložení chyby. Instanci lze podle potřeby rozšířit o metody pro ukládání dalších typů zpráv. Třída pro zpracování přistupuje dále k instanci, která obsahuje informace o všech zařízeních firmy Zeva. Jakým způsobem jsou informace získány, je popsáno v této kapitole dále. Některé informace jsou na základě zpracovávaných zpráv aktualizovány a později poskytovány uživatelům na webových stránkách. Při zpracování se například aktualizuje informace o poslední odezvě zařízení nebo informace o jeho poruchovosti. Při zpracování je nejdříve zjištěn typ zprávy. Pokud se jedná o chybovou zprávu je testováno, jestli je hodnota na pátém bajtu "BB". Tato hodnota udává, že byla u zařízení odstraněna závada. Na základě tohoto zjištění je aktualizována informace konkrétního zařízení. Pokud hodnota na pátém bajtu není "BB", jedná se o novou chybu zařízení. V tom případě je chyba uložena do databáze a je aktualizována informace o zařízení. Stav zařízení je v tomto případě po aktualizaci chybový. Při chybě je dále vytvořena zpráva pro notifikaci, která je zaslána pomocí modulu notifikace uživatelům. Uživatelé musí mít samozřejmě nastaveny dané notifikace v systému. Další zpráva firmy Zeva, u které může dojít ke zpracování, je testovací zpráva. Testovací zpráva obsahuje mimo jiné informaci o tom, zda je zařízení v chybovém stavu či nikoliv. Na základě těchto informací se změní udržované informace o zařízení. To se může zdát jako duplicitní, neboť stejné informace se již nastavují při zpracování chybové zprávy. Chybová zpráva je však zasílána pouze při chybě nebo opravě zařízení. Při výpadku serveru jsou nastavené informace ztraceny. Pomocí testovacích zpráv je možné zachovat integritu informací i po výpadku serveru. Kromě aktualizace informací o chybovosti za-

řízení je aktualizována při zpracování testovací zprávy také informace o poslední odezvě zařízení. Po přijetí zprávy v REST modulu probíhá tedy zpracování následovně. Je vytvořena instance konkrétní zprávy. Po zpracování je zjištěno, jestli vytvořená instance zprávy implementuje rozhraní s metodou pro navrácení zprávy převodníku. Pokud ano, vytvoří se textová verze potvrzovací zprávy, která je odeslána převodníku a zpracování v tomto okamžiku končí.

K definici informací o zařízeních slouží v sytému rozhraní, které definuje informace zmíněné výše - poslední odezva zařízení, chybovost zařízení. Kromě nich definuje také ID, IP adresu a port zařízení. Poslední dvě zmíněné vlastnosti jsou využívány pro vzdálené nastavení či zaslání parametrů. Pro informace o zařízeních je vytvořena kolekce objektů. C# umožňuje vytvoření tzv. konkurentních kolekcí. Jejich používání se od obyčejných kolekcí mírně liší. Tyto kolekce je možno sdílet mezi více vlákny. To je důležité, neboť s informacemi o zařízeních pracují instance tříd v oddělených vláknech. Rozhraní s informacemi implementuje v systému třída s REST informacemi. Ta doplňuje k definovaným informacím URL adresu převodníku jak pro nastavení, tak pro zaslání parametrů. Kolekci s informacemi o zařízeních uchovává třída pro vzdálené zpracování parametrů. Tato třída využívá informace k nastavení či zaslání vzdálených parametrů. Pro tyto účely implementuje třída rozhraní s metodami - zašli parametry a nastav parametry. Dále implementuje třída rozhraní, které definuje metody pro zacházení s kolekcí. Jedná se například o metodu pro přidání zařízení do kolekce. Dále metody pro nastavení a odebrání poruchového stavu zařízení a aktualizování poslední odezvy zařízení, které jsou využívány pro aktualizaci informací při zpracování zpráv přijatých z převodníku. Další metody, které rozhraní definuje, jsou využívány pro zobrazování informací na webu. Jedná se o metody, které vrací poslední odezvu konkrétního zařízení a metodu pro výpis všech poruchových zařízení. Kolekce s informacemi o zařízeních je po inicializaci systému prázdná a jsou do ní dynamicky přiřazovány zařízení v třídě TCP serveru. Proto třída TCP serveru přebírá v konstruktoru instanci této kolekce. TCP server slouží k získávání některých informací o připojených zařízeních. Další informace jsou doplněny při zpracování zpráv. V Kapitole 5.3.2 se nachází popis zasílání informací o zařízení převodníkem. Data jsou přenášeny ve formátu JSON. Po přijetí server data deserializuje a uloží do zmiňované kolekce. Ke kolekci se přistupuje napříč celým systémem. Pro účely diplomové práce bylo zapůjčeno pouze jedno zařízení, se kterým je systém testován. Proto při inicializaci systému vznikají některé instance zařízení, aniž by byl použit k jejich vytvoření TCP server. Tímto způsobem lze zobrazovat na webu "falešná"zařízení. Samozřejmě je nutné, aby měla zařízení validní informace v databázi.

Pro komunikaci s převodníkem slouží rozhraní, které definuje metody pro vzdálené nastavení a přijetí parametrů a metodu pro přijetí informací o zařízení. Metoda pro přijetí informací o zařízení slouží pro získání dat, které nejsou uchovávány v kolekci. Touto metodou jsou získávány informace z řídicí jednotky, které nejsou zasílány na server, ale jsou ukládány do paměti převodníku. Rozhraní pro komunikaci s převodníkem, je využíváno ve webové části, která bude popsána následovně. Protože rozhraní abstrahuje implementaci, změna realizace komunikace s převodníkem by znamenala pouze změnu implementace definovaných metod. Metody definované v rozhraní pro komunikaci s převodníkem jsou v diplomové práci realizovány pomocí REST architektury. Informace potřebné k zaslání REST dotazu jsou získávány z kolekce s informacemi. V případě zaslání REST dotazu konkrétnímu zařízení jsou podle ID získány informace z kolekce a je asynchronně zaslán dotaz. Pro větší obecnost je vždy vrácena HTTP zpráva. Z HTTP zprávy lze získat informaci o kódování zprávy a její obsah je možné deserializovat na instance tříd.

5.4.4 Webový modul

Webový modul serverové části slouží pro vyřizování požadavků klienta. K tomuto účelu je zejména využíváno databázového modulu a modulu pro komunikaci s převodníkem. Webový modul je realizován pomocí .NET frameworku, který poskytuje rozšířené vlastnosti REST API serveru. Jedná se například o validaci příchozích dat klienta nebo poskytování instancí, potřebných k fungování webového modulu. K vyřizování požadavků klienta jsou definovány 3 kontroléry. Autentizační kontrolér, sloužící pro registraci a přihlašování uživatele, kontrolér uživatele, který slouží například pro poskytování informací o uživateli a přidávání notifikací. V posledním kontroléru pro Zeva zařízení se nachází veškeré operace, týkající se zařízení firmy Zeva. Pomocí tohoto kontroléru lze zpracovat požadavek na vzdálené nastavení či přijetí parametrů. Server webového modulu naslouchá na specifikované adrese (v rámci diplomové práce je to http://localhost/api s portem 5000). K adrese se přidává název kontrolérů, což definuje cestu k metodám daného kontroléru. Poslední částí adresy jsou konkrétní metody s parametry, které vyřizují požadavky uživatele. Dále konkrétní metody specifikují, jaký typ REST API požadavků vyřizují. Adresa metody pro získání parametrů ze zařízení může vypadat takto: `http://localhost:5000/zevaDevice/getParameter/1`. Část zevaDevice označuje název kontroléru a getParameter konkrétní metodu, která obstarává vyřízení požadavků pro zaslání parametrů konkrétního zařízení. Číslo na konci adresy udává zjednodušený příklad identifikátoru zařízení, který je předán do parametru metody. U metod, které jsou specifikované pro HTTP POST dotazy, klient zasílá na danou adresu JSON objekt. V metodě je jako parametr specifikována třída, která odpovídá struktuře JSON objektu. Ve třídě je možné specifikovat, jakých hodnot mohou nabývat jednotlivé atributy třídy. Po přijetí dotazu je objekt automaticky převeden na instanci a je provedena kontrola, zdali je obsah všech atributů validní. Pokud obsah atributu validní není, v kontextu dotazu je udržována informace o chybě. Po převedení JSON objektu se s instancí třídy dále pracuje. Pro vyřizování požadavků kontroléry potřebují ostatní moduly serveru. Při každém novém požadavku vzniká nová instance kontroléru. Přístup k databázi je řešen pomocí singletonu, který lze použít ve všech částech systému. Poskytování ostatních potřebných instancí zařizuje .NET framework za pomocí techniky vkládání závislostí[7]. Při inicializaci webového modulu je specifikováno, které instance má .NET framework poskytovat. Jednotlivé kontroléry v konstruktoru obsahují rozhraní, do kterých jsou při vytvoření nové instance kontroléru vkládány potřebné závislosti. Tímto způsobem jsou v kontrolérech získány například informace o zařízeních a instance sloužící pro komunikaci s převodníkem. Pro větší přehlednost a obecnost jsou databázové požadavky vyřizovány pomocí instancí implementující rozhraní "repositářů". V instancích konkrétních repositářů jsou specifikovány metody, které přistupují k požadovaným modulům a vracejí data. Klientovi jsou data vrácena jako objekt, ve kterém jsou specifikovány všechny informace. Webový modul tyto objekty definuje. Protože získání všech potřebných dat nemusí být vyřízeno v jednom kroku, je nutné do objektu data postupně přidávat. Pro přidávání dat do objektů byl v diplomové práci použit "AutoMapper". S pomocí něho se nemusí mapovat hodnoty atributů objektů postupně. Instanci "AutoMapperu"lze specifikovat, které objekty mají mezi sebou vztah. V diplomové práci mají mapované atributy objektů vždy stejný název. Lze však specifikovat i konkrétní atributy, které se mají či nemají mapovat. Klientovi jsou vráceny podle typu dotazu HTTP zprávy. V popisu webového modulu nebude uveden způsob vyhodnocení všech metod. Bude zde popsán pouze způsob, jakým je implementováno přihlášení

[7]Vysvětlení techniky vkládání závislostí je dostupné na: https://docs.microsoft.com/cs-cz/aspnet/core/fundamentals/dependency-injection

uživatele, neboť souvisí s autorizací uživatelů v celém systému. Některé kroky při zpracování přihlášení uživatele jsou podobně prováděny i v ostatních metodách, a proto je může následující popis charakterizovat také.

Pro přihlášení uživatele je v kontroléru uvedena metoda s definovanou URL adresou vyřizující POST požadavky. V parametru přijímá objekt přihlášení s atributy loginu a hesla uživatele. Po přijetí požadavku a zkontrolování atributů se v metodě pracuje s instancí uchovávající výše zmíněná data. V první fázi se pomocí repositáře zkontroluje, zdali v databázi existuje uživatel se zadaným loginem a heslem. Hesla uživatelů jsou v databázi šifrované a veškerá logika zpracování je zapouzdřena v metodě repositáře. Pokud je repositářem vrácena prázdná instance, uživatel v databázi neexistuje a přihlášení je neplatné. V tom případě je zpracování přihlášení ukončeno a klientovi je vrácena HTTP zpráva s kódem informujícím o špatném přihlášení (401 - Unauthorized). Pokud je repositářem vrácen záznam o uživateli, vytváří se JWT (JSON Web Token). JWT token je standard definující kompaktní způsob pro zabezpečený přenos informací mezi dvěma stranami [3]. Tato informace obsahuje digitální podpis. Podpis může být proveden buď pomocí tajného hesla HMAC algoritmem, anebo pomocí sdíleného/privátního klíče za použití RSA algoritmu. V této diplomové práci je použit algoritmus HMAC. Do JWT tokenu lze přidat určité informace. Při vytváření tokenu uživatele jsou specifikovány informace, jako jsou ID uživatele, jméno uživatele, role uživatele. Tyto informace jsou získány z databáze při kontrole loginu a hesla. Kromě toho se uvádí expirační doba tokenu, která je v této diplomové práci jednodenní. Heslo potřebné k uložení tokenu je uvedeno v konfiguračním souboru. Vytvořený token je vrácen klientovi. Token musí být klientem zaslán při každém dotazu, kde je potřebná jeho autorizace.

5.5 Klient

Logika klienta byla v diplomové práci implementována za použití frameworku Angular a jazyku Typescript. K vizualizaci stránek byly použity jazyky HTML, CSS a framework Boostrap.

Framework Angular definuje komponenty, které obsahují HTML, CSS a Typescript soubory. Komponenty se využívají k implementaci konkrétní stránky. HTML a CSS soubory slouží pro vizualizaci stránek. V Typescript souboru se nacházejí informace poskytnuté serverovou částí, instance modulů s určitou funkčností, a je zde implementována logika stránky. Z HTML souborů lze přistupovat pomocí zvláštní syntaxe k informacím definovaných v Typescript souborech stejné komponenty. Pro získání informací ze serveru Angular definuje tzv. služby. Metody služeb definují HTTP dotazy, které jsou zasílány na danou adresu webovému modulu serverové části. Služby lze využívat pro získání informací buď přímo v Typescript komponentě nebo v tzv. resolveru. Resolvery slouží k zautomatizování poskytování informací Typescript komponentě. Využívají služeb pro získání informací ze serverové strany. Spouštějí se na základě URL adresy, na které se uživatel nachází a mohou zasílat službám parametry, které jsou z URL adresy získány. V adresáři klienta se nachází soubor s tzv. routami. V tomto souboru je přiřazena konkrétní komponenta pro vizualizaci stránky k dané URL adrese. Pokud uživatel zadá adresu, která není specifikována, je přesměrován na adresu, definovanou na konci souboru pomocí wild card masky. Komponenty mohou volat služby k získání informací ze serveru interně v Typescript souborech. Dalším způsobem je definovat resolver ke konkrétní URL adrese. Pokud uživatel přistoupí k URL adrese, ke které je přiřazena komponenta a resolver, v Typescript souboru komponenty budou vždy aktuální data.

Ve výše uvedeném popisu byl stručně představen způsob fungování Angular frameworku a dalších součástí na straně klienta. Tímto způsobem je implementována celá logika klienta. Každá stránka má přiřazenou komponentu, která zajišťuje její funkčnost a vizualizaci. Jednotlivé komponenty si dále mohou v případě potřeby určitým způsobem předávat data. V systému také lze používat již vytvořené moduly poskytující různou funkčnost. Může se jednat například o "spinner", který je v diplomové práci zobrazen, pokud uživatel čeká na data nebo "slider"používaný k zadání hodnot při nastavování vzdálených parametrů. Tyto závislosti byly instalovány pomocí NPM nástroje. Velké množství modulů již obsahuje Angular framework po instalaci. Může se jednat například o formulářové prvky s možností validace. Po přihlášení uživatele a následném vrácení tokenu ze strany serveru je token uložen do paměti prohlížeče. Pomocí modulu dodaného NPM instalací je token zasílán při každém zaslání HTTP dotazu. Podoba a funkčnost jednotlivých stránek bude popsána v další kapitole.

Kapitola 6

Testování

Testování systému bylo prováděno za použití testovacího kitu zapůjčeného firmou Zeva. Některé chyby testovacího kitu byly zjištěny již před testováním. Testovací modul nesimuluje otevření dveří po dojetí do patra. Nejprve byla zkoušena pouze základní komunikace mezi převodníkem a řídicí jednotkou. Testovalo se přijímání chybových, testových zpráv a zpráv s počítadly. Pomocí tlačítek a spínačů na testovacím modulu byly zasílány převodníku požadované zprávy. Bylo zkoumáno, jestli se v převodníku vytvářejí správné instance zpráv, a jestli zprávy obsahují požadované data. Dále bylo pozorováno, jestli jsou zprávy z řídicí jednotky správně potvrzovány převodníkem. To lze vypozorovat na displeji řídicí jednotky, neboť při nepotvrzení zprávy na displeji začne blikat symbol "E". Při tomto testu byla vypozorována chyba. Zpráva s počítadly obsahuje měsíční hodnoty vždy nulové. Tato chyba byla diskutována s pracovníky firmy, kteří pracují na její nápravě a na vydání nového firmwaru. Po otestování funkcionality základních zpráv byla kontrolována funkčnost vzdáleného nastavení a zasílání parametrů. Funkčnost byla kontrolována s použitím nástroje Postman. Postman umožňuje zasílání libovolných REST API dotazů na specifikovanou adresu. Pomocí tohoto nástroje lze přizpůsobit dotaz podle požadavků. Po zadání URL adresy si lze vybrat typ dotazu. Dále je možné specifikovat autentizační token a další vlastnosti. Výsledek dotazu je v tomto nástroji přehledně zobrazen. Nejprve byla testována funkčnost vzdáleného zaslání parametrů řídicí jednotkou. K tomuto účelu byl nástrojem Postman zaslán na specifikovanou adresu dotaz GET. Výsledkem dotazu je objekt ve formátu JSON, který je po úspěšném dokončení zobrazen v nástroji. Veškeré hodnoty v navráceném objektu byly porovnávány s hodnotami specifikovanými firmou, které je možné vidět v příloze A. I zde byla objevena chyba, neboť ve druhé konfigurační zprávě byla hodnota v registru D1 mimo definovaný rozsah. Po dohodě s pracovníky firmy byla hodnota pouze přenastavena a správná hodnota bude zanesena do nového firmwaru řídicí jednotky, ve kterém budou opraveny všechny zjištěné chyby. Po otestování přijetí vzdálených parametrů bylo ověřováno vzdálené nastavení parametrů. Pro nastavení byl použit HTTP dotaz POST, do jehož těla byl textově přidán objekt se všemi potřebnými parametry. V případě platného nastavení parametrů lze na displeji řídicí jednotky sledovat symbol "Z", který informuje, že je řídicí jednotka připravena na zápis parametrů. Po potvrzení parametrů byly opětovně znovu vzdáleně zaslány parametry řídicí jednotky a sledovány změny. Při tomto testu nebyly nalezeny žádné další chyby. Dále bylo testováno paralelní nastavení a přijetí parametrů. Pokud byl testován nepovolený současný přístup k vzdáleným operacím (například současný zápis parametrů), převodník reagoval očekávaně - chybou. Při povoleném přístupu, tedy paralelním přijetím parametrů, se systém choval odlišně než bylo žádáno. První požadavek byl vždy odbaven separátně, ostatní požadavky byly vyřízeny paralelně. Po důkladném zkoumání

tato chyba nebyla odstraněna. Chyba neomezuje funkčnost převodníku a při paralelním požadavku na zaslání dat jsou veškeré požadavky vyřízeny maximálně po 1 minutě. Chyba bude dále konzultována s pracovníky firmy.

Po otestování funkčnosti převodníku a komunikace mezi řídicí jednotkou a převodníkem, probíhal test serverové části. Zde byly nejprve separátně otestovány jednotlivé moduly, neboť byly také separátně vyvíjeny. Konkrétní modul byl vždy po dokončení vyzkoušen v hlavní části programu. U modulu databáze byly jednotlivé dotazy nejprve vyzkoušeny v programu Microsoft SQL Management Studiu. Poté byla funkčnost testována zavoláním jednotlivých metod konkrétních instancí a sledováním změny databáze ve stejném programu. U modulu pro notifikace byly při inicializaci načteny všechny notifikace z databáze. Hned po inicializaci se vytvořila instance zprávy pro modul notifikace, která byla předána v parametru metody notifikuj. Po vykonání metody bylo zkoumáno, zdali se zaslaly všechny notifikace na nastavená zařízení. Modul pro zpracování zpráv potřebuje všechny části, jejichž popis testování byl uveden, neboť využívá databázi i modul notifikace. Modulu pro zpracování zpráv byly postupně předávány jednotlivé instance zpráv a bylo sledováno jejich zpracování. Tento modul byl posléze integrován do modulu pro komunikaci s převodníkem. V tento moment již server zpracovával a vyhodnocoval veškeré zprávy z řídicí jednotky. Poté se otestované části postupně zapracovávaly do webového modulu. Jednotlivé části webového modulu včetně vzdáleného nastavení a zaslání parametrů se opět testovaly za použití nástroje Postman. Po otestování webového modulu bylo možné zasílat dotazy webovému modulu klientem. Veškeré testy probíhaly ve webovém prohlížeči, kde se sledovalo požadované chování. Pokud byla nalezena chyba, nejprve se použil nástroj Postman pro zaslání dotazu na server. Pokud byl dotaz zpracován v pořádku, chybu bylo nutné najít a odstranit na straně klienta. Ve finální verzi programu přetrvává pouze již popisována chyba při paralelním získání parametrů, která však neovlivňuje funkcionalitu. Další chyby se ve finální verzi programu nevyskytly.

Další fází testování bude po konzultaci s pracovníky firmy nasazení systému do reálného provozu. Nejprve bude systém otestován pouze na jediném výtahu. Při úspěšném testu lze systém nasadit na větší počet výtahů.

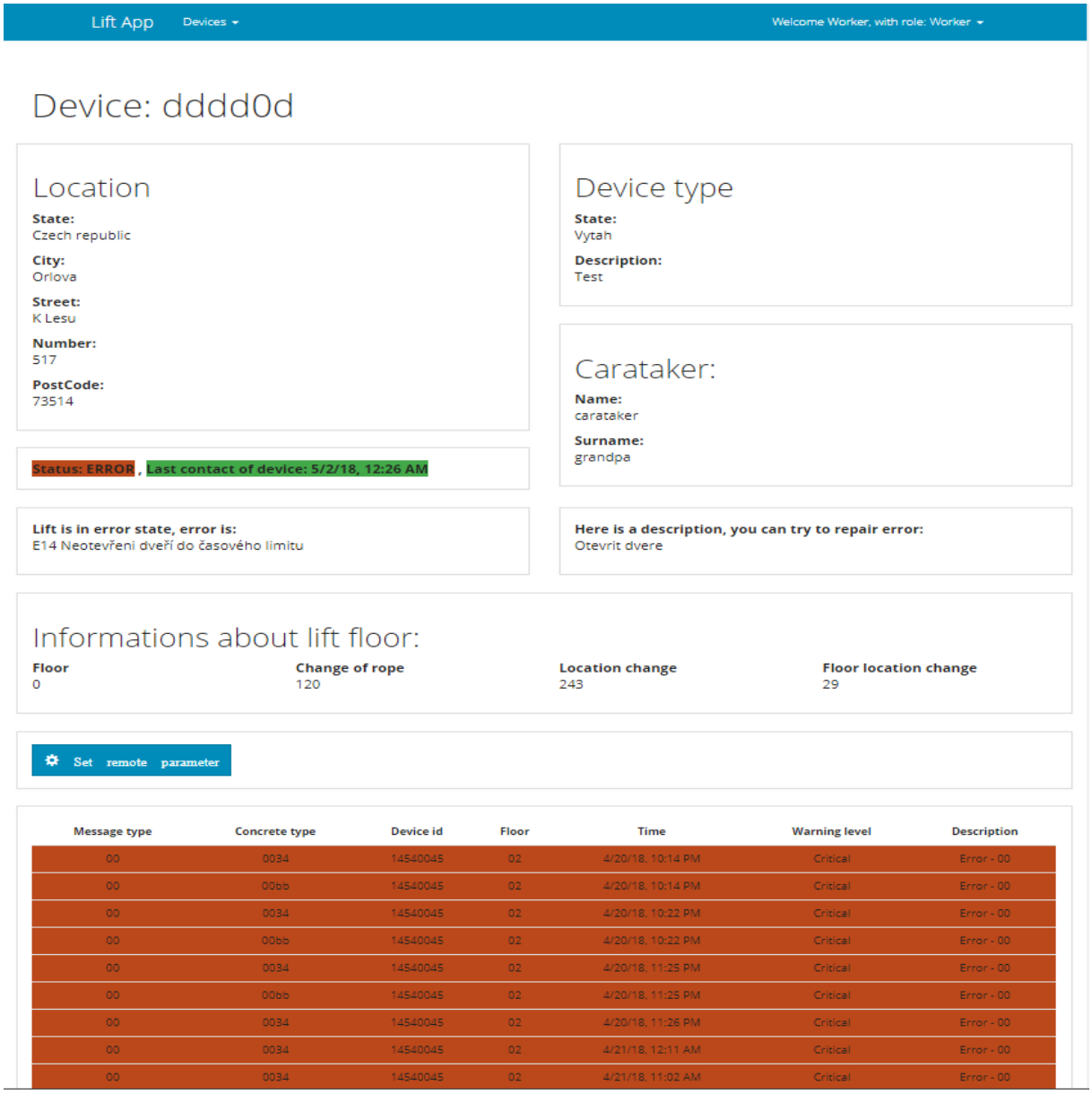
Kapitola 7

Podoba systému

Systém nabízí pro uživatele několik stránek. Po přihlášení je uživatel přesměrován na domovskou stránku. Navigační panel v horní části obrazovky je zobrazován na všech stránkách. Podle úrovně autorizace jsou v navigačním panelu uživateli zobrazovány jeho možnosti. S účtem s pravomocí administrátora lze pouze vytvářet nové uživatelé systému a vypsat si informace o všech uživatelích. Systém kontroluje, jestli jsou při vytváření zadána pouze validní data. Pokud jsou zadána nevalidní data (například již existující login či špatně zadané heslo), uživatele nelze vytvořit. Uživateli je přiřazována role. Při nastavení role technika lze uživateli přiřadit také firmu. Pro nastavení rolí a firmy systém poskytuje administrátorovi možnosti, ze kterých si vybírá. Nelze tedy například přiřadit technikovi firmu, která v systému neexistuje. Administrátor nemá pravomoc k nahlížení na zařízení jiných firem.

Uživatel přihlášen pod účtem s rolí domovníka má v navigačním panelu volbu "zařízení". Po kliknutí na tuto položku mu jsou zobrazeny možnosti filtrování. Domovník si může zobrazit buď všechny zařízení nebo všechny poruchové zařízení. Domovníkovi se zobrazují pouze zařízení, které mu byly přiřazeny. Domovník může nahlédnout na detail zařízení. V tomto okně mu nejsou zobrazeny některé informace, které jsou přístupné pouze technikům, kteří dané zařízení spravují.

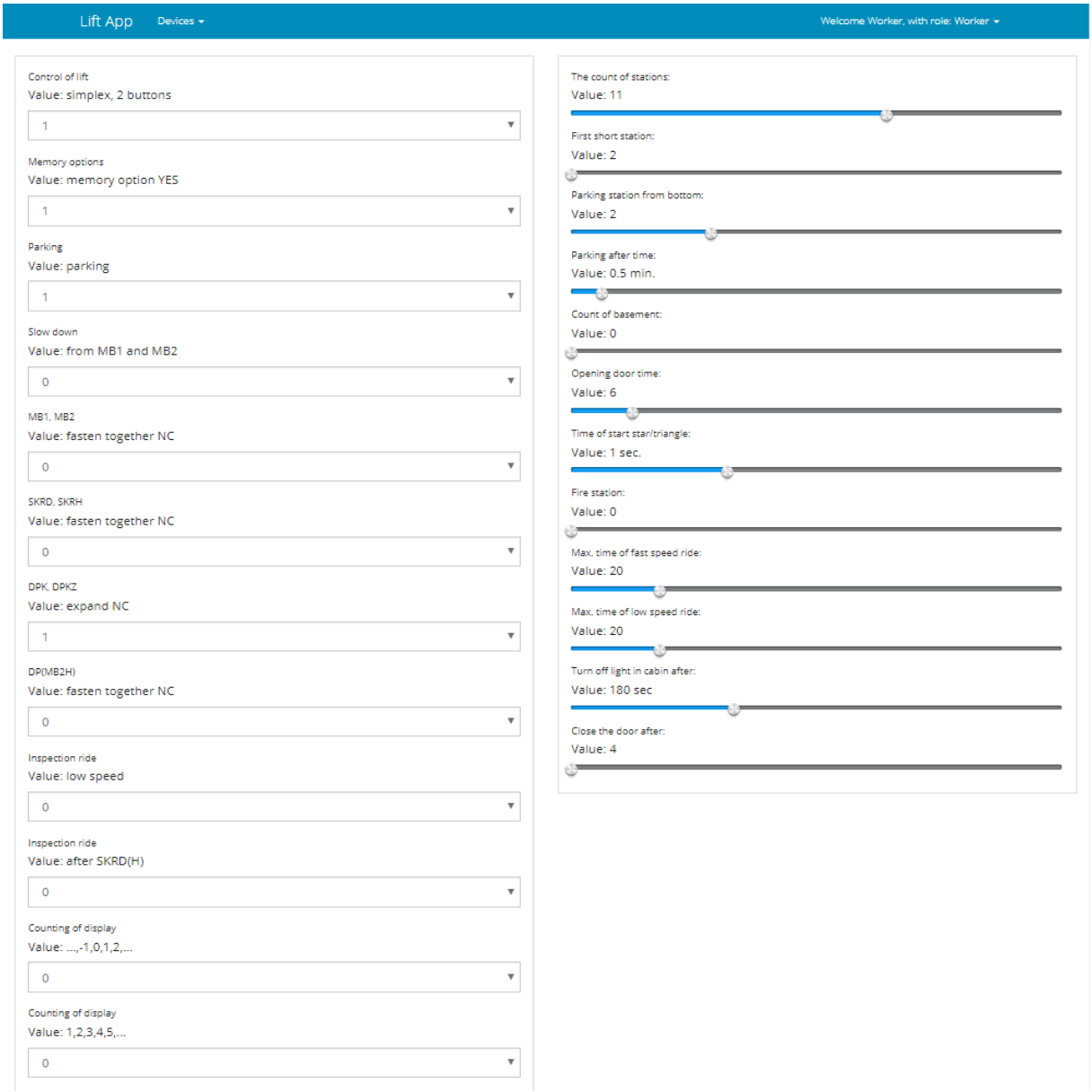
Nejvíce možností v systému má technik, tedy pracovník firmy. Po přihlášení pod účtem technika lze v editaci profilu spravovat notifikace. Uživateli jsou vypsány všechny nastavené notifikace. Pro vytvoření nové notifikace je uživatel přesměrován na novou stránku. Lze nastavit typ notifikace a úroveň závažnosti zprávy, při kterém se notifikace zašle. Notifikace lze zasílat buď při každé chybě nebo při konkrétní chybě firmy. Technik má více možností filtrování zařízení než domovník. Tak jako domovník má i technik možnost zobrazit si všechny zařízení a všechny poruchové zařízení. Dále má technik možnost filtrovat zařízení podle typu. Technikovi jsou po zvolení konkrétního typu zobrazeny například všechny výtahy firmy. Poslední možnosti technika je filtrování podle lokality. Technikům jsou zobrazeny veškeré lokality, na kterých firma spravuje zařízení. Po zvolení konkrétní lokality jsou technikům vypsány například všechny zařízení v panelovém domě na vybrané adrese. Do systému lze v budoucnu po dohodě s pracovníky firmy integrovat další možnosti filtrování. V popisu diplomové práce bude ukázána pouze podoba složitějších stránek. Ukázky ostatních stránek jsou přiloženy v příloze B. Pokud si technik zobrazí detail výtahu, je mu zobrazena stránka, kterou je možné vidět na obrázku 7.1.



Obrázek 7.1: Detail zařízení

V horní části je vidět zmiňovaný navigační panel, který je zobrazován na všech stránkách. V pravé části panelu je vypsáno jméno přihlášeného uživatele a jeho role. Po kliknutí na jméno uživatele je možné editovat jeho profil a spravovat notifikace. Pod navigačním panelem je vidět ukázka stránky s detailem zařízení. Pro přehled je zobrazen identifikátor zařízení. Pod ním jsou uvedeny další informace. Jedná se o lokaci a typ zařízení. Pod lokací se nachází status zařízení a poslední kontakt zařízení. Výše zmíněné informace vidí v detailu také domovník. Další zobrazené informace vidí pouze technik. Pokud je k zařízení přiřazen domovník, je jméno a příjmení domovníka zobrazeno. V případě chyby je technikovi zobrazena konkrétní chyba s návodem na opravu. Ve spodní části detailu je také zobrazena historie všech chyb zařízení. Pokud je kontakt se zařízením v pořádku, nachází se na stránce informace o počítadlech. V opačném případě jsou načteny všechny informace o zařízeních,

ale kolonka s počítadly informuje, že tyto informace nejsou k dispozici. O přerušeném kontaktu informuje také informace o posledním kontaktu zařízení. Pokud ze zařízení není do tří minut přijatá testovací zpráva, tato informace pro větší přehlednost zčervená. Na obrázku je vidět modré tlačítko, které slouží pro vzdálené nastavení parametrů. Při vzdáleném nastavení parametrů jsou nejprve uživateli zobrazeny aktuální parametry zařízení. Proto je tlačítko v systému používáno také pro vzdálené přijetí parametrů. Po kliknutí na tlačítko je uživateli zobrazena stránka, kterou lze vidět na obrázku níže.



Obrázek 7.2: Nastavení a přijetí parametrů zařízení

V levé části stránky jsou všechny konfigurační parametry registrů B0-Bf. Pro velký počet parametrů registrů B0-Bf jsou na obrázku ukázány pouze některé z nich. V pravé části se nacházejí konfigurační parametry registrů D0-Db. Uživateli jsou veškeré nastavitelné hodnoty poskytnuty k výběru a jsou zobrazovány uživatelsky příjemným způsobem. U parametrů s registry B0-Bf si lze konkrétní parametr vybrat podle jeho významu. U konfiguračních parametrů s registry D0-Db jsou uživateli zobrazeny hodnoty násobené definovaným koeficientem. Vysvětlení zobrazení těchto parametrů je uvedeno v příloze A. Uživatel si hodnoty registrů D0-Db vybírá pomocí posuvníku. Tímto způsobem může uživatel nastavit pouze validní data. Posuvník hlídá hraniční hodnoty jednotlivých registrů. Pokud registr může obsahovat pouze hodnoty s určitým krokem (např. 2, 4, 6 ...) má posuvník krok nastaven. Veškeré hodnoty jsou před odesláním na server převedeny.

Protože nastavení i přijetí parametrů trvá delší dobu je při čekání uživateli zobrazen spinner s textem, informujícím o probíhající operaci. Uživatel je poté informován o výsledku operace. Informování uživatele probíhá pomocí zpráv v pravé části obrazovky. Tímto způsobem jsou uživateli zobrazovány také chyby. Chyba se může objevit například pokud se serverové části nepodaří zpracovat požadavek. Jestliže dojde k neplatnému zadání loginu či hesla, do formulářů jsou zadány nevalidní data nebo uživatel není autorizovaný k přístupu.

Kapitola 8

Závěr

Cílem této práce bylo vytvořit systém pro vzdálenou správu výtahů. Implementace tohoto systému je primárně určena pro pracovníky firmy Výťahy Zeva.

Firmou Výťahy Zeva byla poskytnuta řídicí jednotka, která neobsahovala žádný mechanizmus pro zasílání zpráv na server. Proto bylo nutné vybrat vhodný převodník a implementovat systém, který zpracovává zprávy zaslané řídicí jednotkou. Systém se podařilo úspěšně implementovat, až na chybu, která je popsána v kapitole 6. Tato chyba vzniká při paralelním požadavku na zaslání parametrů. Parametry jsou uživateli vždy zaslány a chyba tak nemá velký vliv na funkcionalitu. Chyba bude probrána s pracovníky firmy Zeva a v budoucnu opravena.

Realizace diplomové práce pokračovala implementací serverové části. Serverovou část se podařilo realizovat téměř do formy, která byla stanovena na začátku diplomové práce. Projekt je implementován do takové fáze, aby ho bylo možno pracovníkům firmy představit. Již nyní jsou naplánovány některé rozšíření. Pro pracovníky firmy by bylo jednodušší konfigurovat veškeré aspekty systému přímo ve webovém prohlížeči. Pokud se podíváme na diagram užití v kapitole 3, diplomová práce umožňuje většinu požadovaných vlastností vykonávat z webového prohlížeče. Administrátorovi zatím nebyla poskytnuta možnost vytvoření nové firmy a možnost přiřazení domovníka k výtahu. Do systému budou později přidány operace, které administrátorovi umožní firemní nastavení celého systému. Může se jednat například o vytvoření všech druhů chyb firmy, úrovně závažnosti zpráv, typů zařízení apod. Nyní jsou tyto věci vkládány do databáze pomocí nástroje Microsoft SQL Management Studio. Server už je na doplnění těchto vlastností připraven. Výše uvedené změny je plánováno do systému zanést v nejbližší době. Později lze do systému přidat technikům u konkrétních zařízení poznámky a mapu lokace zařízení. Pro tyto změny už je připravena databáze. I po těchto úpravách bude mít systém spoustu možností pro rozšíření. Kromě šifrovaných hesel v databázi a autorizace pomocí JWT tokenu nebyla v rámci diplomové práce řešena bezpečnost. Fatální narušení bezpečnosti hrozí pouze v případě zneužití možnosti vzdáleně nastavit parametry. Parametry může v systému změnit pouze autorizovaná osoba. Pokud by útočník znal protokol řídicí jednotky a IP adresu převodníku, mohl by teoreticky nastavit výtahu libovolné parametry. Po nastavení je však nutné technikem ručně potvrdit změnu parametrů. V budoucnosti by se do systému mohla přidat dostatečná úroveň zabezpečení a nebyla by nutná přítomnost technika při nastavení parametrů. Možnosti jak lze zabezpečení realizovat se nacházejí v kapitole 4.2. Další případnou změnou, která by mohla přinést pracovníkům lepší přehled o zařízeních, je přidání statistik. Systém by mohl například vizualizovat počty chyb zařízení v jednotlivých měsících v podobě grafu. Pro správu velkého množství zařízení by bylo možné kód více optimalizovat a některé poža-

davky paralelizovat. Kritické jsou zejména databázové operace, ke kterým systém přistupuje neustále. Bylo by užitečné, kdyby systém poskytoval pracovníkům různé predikce. Systém by se mohl učit z chyb zařízení a na základě nasbíraných informací například upozornit na možný výskyt další chyby v nejbližších dnech. Do systému kromě vzdáleného monitoringu zařízení lze později přidat i další funkcionalitu či jiná zařízení. Jednou z možností rozšíření je RFID technologie pro identifikaci objektů. Návrh možných rozšíření je popsán v kapitole 4.5.

V bakalářské práci byla realizací převodníku pověřena firma Camea. Komunikace mezi firmou Camea a firmou Výťahy Zeva neprobíhala optimálně a převodník se firmě Camea nepodařilo zprovoznit. V Bakalářské práci proto musel být datový provoz simulován. Firmou byly specifikovány další požadavky, jejichž rozšíření v bakalářské práci by bylo problematické. Proto byl proveden nový návrh systému vzdálené diagnostiky. Podle nového návrhu byla tato diplomová práce implementována. Systém oproti bakalářské práci poskytuje zákazníkům více možností. Jedná se například o možnost autorizace v systému. Zařízení lze vybírat pomocí definovaných filtrů. Je možné vzdáleně nastavovat a přijímat parametry výtahu. Pro rychlejší upozornění techniků na vzniklou chybu si mohou pracovníci nastavovat notifikace. Návrh systému je obecnější a další změny nebude tak obtížné implementovat. Databáze je přizpůsobena k tomu, aby bylo možné mít v systému i jiné druhy zařízení než jsou výtahy. Při detailním přehledu zařízení je poskytováno větší množství informací. Převodník byl v rámci diplomové práce realizován a tak může být systém testován v reálném provozu.

Některé z výše zmíněných rozšíření budou do systému přidány co nejdříve. I bez nich může být produkt představen firmě. Po konzultaci s pracovníky může být systém postupně nasazován do produkčního prostředí. Protože byl systém testován s reálnou řídicí jednotkou, neměly by se při nasazení do provozu vyskytnout žádné neočekávané chyby. Pokud se v systému naleznou, bude usilováno o jejich co nejrychlejší nápravu. Systém bude ve spolupráci s firmou Výťahy Zeva dále rozvíjen.

Literatura

[1] *Abstract Factory*. [Online; navštíveno 11.4.2018].
URL http://www.dofactory.com/net/abstract-factory-design-pattern#rea

[2] *Elevator control system*. [Online; navštíveno 13.11.2017].
URL http://elevation.wikia.com/wiki/Elevator_control_system

[3] *Introduction to JSON Web Tokens*. [Online; navštíveno 25.4.2018].
URL https://jwt.io/introduction/

[4] *Observer*. [Online; navštíveno 17.4.2018].
URL http://www.dofactory.com/net/observer-design-pattern

[5] *Rekuperační jednotka REVYT*. [Online; navštíveno 29.11.2017].
URL http://moderni-vytahy.cz/cs/co-delame/vytahove-komponenty/rekuperace.html

[6] *Srovnání VPN Protokolů: PPTP vs. L2TP vs. OpenVPN vs. SSTP vs. IKEv2*. [Online; navštíveno 7.1.2018].
URL https://cs.vpnmentor.com/blog/srovnani-vpn-protokolu-pptp-vs-l2tp-vs-openvpn-vs-sstp-vs-ikev2/

[7] *Stopařův průvodce REST API*. [Online; navštíveno 10.4.2018].
URL http://www.itnetwork.cz/nezarazene/stoparuv-pruvodce-rest-api

[8] *What is a serial to Ethernet converter and how does it work?* [Online; navštíveno 10.1.2018].
URL http://www.usconverters.com/index.php?main_page=page&id=70&chapter=0

[9] *Basic Elevator Components*. 2013, [Online; navštíveno 12.11.2017].
URL http://www.electrical-knowhow.com/2012/04/basic-elevator-components-part-two.html

[10] *VÝTAHY, s.r.o.* 2014, [Online; navštíveno 06.11.2017].
URL http://www.vytahy.com/cs/vyroba-a-modernizace-vytahu/nakladni-vytahy

[11] *MP lift*. 2015, [Online; navštíveno 06.11.2017].
URL http://www.mplifts.cz/budova-bez-vytahu/

[12] *Solární výtahy mají budoucnost*. 2016, [Online; navštíveno 1.1.2018].
URL http://www.konstrukce.cz/clanek/solarni-vytahy-maji-budoucnost/

[13] *ThysenKrupp*. 2018, [Online; navštíveno 06.11.2017].
URL http://thyssenkrupp-vytahy.cz/products/elevators/

[14] Apter, R.; Prathaler, M.: Regeneration of power in hybrid vehicles. In *Vehicular Technology Conference. IEEE 55th Vehicular Technology Conference. VTC Spring 2002 (Cat. No.02CH37367)*, IEEE, doi:10.1109/vtc.2002.1002987, [Online; navštíveno 22.11.2017].
URL http://ieeexplore.ieee.org/document/1002987/

[15] Misran, M.; Said, M. M.; Othman, M.; aj.: Design and development of RFID based Elevator. In *2014 International Symposium on Technology Management and Emerging Technologies*, IEEE, may 2014, doi:10.1109/istmet.2014.6936515, [Online; navštíveno 18.11.2017].
URL http://ieeexplore.ieee.org/document/6936515/

[16] (red): *Výtahy s rekuperací pro budovu Enterprise*. 2016, [Online; navštíveno 1.1.2018].
URL http://www.konstrukce.cz/clanek/vytahy-s-rekuperaci-pro-budovu-enterprise/

[17] Selmeczy, P.: *VPN vs SSH*. 2014, [Online; navštíveno 7.1.2018].
URL https://www.bestvpn.com/vpn-vs-ssh/

[18] Sit, J. M. J.: *Fastest elevator*. CNN, 2017, [Online; navštíveno 08.11.2017].
URL
http://edition.cnn.com/style/article/worlds-fastest-tower/index.html

[19] Weisser, F.: *VZDÁLENÁ DIAGNOSTIKA PORUCH VÝTAHU*. 2016, [Online; navštíveno 30.3.2018].
URL http://www.fit.vutbr.cz/study/DP/BP.php.cs?id=18548&file=t

[20] Welch, P. J.; Gannet Fleming, I.; Camp Hill, P.: *Remote Monitoring Of Elevators And Escalators: Managing The Alarms And The Maintenance*. [Online; navštíveno 1.1.2018].
URL http://www.vtexcellence.com/docs/VTX_RemoteMonitoring.pdf

[21] Xiao, X.; Wu, H.; Zhang, Y.; aj.: Design of Elevator Intelligent Control System Based on Long Distance RFID in Pervasive Computing Environment. In *2011 International Symposium on Computer Science and Society*, IEEE, červenec 2011, doi:10.1109/isccs.2011.42, [Online; navštíveno 16.11.2017].
URL http://ieeexplore.ieee.org/document/6004283/

Příloha A

Význam chybových kódů a konfiguračních parametrů

Význam chybových kódů

kód chyby	ID chyby a význam
0x20	E0 přerušený obvod 73 při jízdě
0x21	E1 chyba snímání MB1, MB3(4);
0x22	E2 chyba snímání
0x23	E3 současně sepnutý horní a dolní SKRH/SKRH kontakt
0x24	E4 přerušený obvod 75 při jízdě
0x25	E5 překročení doby jízdy ze stanice do stanice
0x26	E6 zaseknutí stykačů hlavního pohonu
0x27	E7 současné nasnímání MB1 a MB2
0x28	E8 výpadek napájení, režim baterie
0x29	E9 Nepodařilo se uzavřít obvod 500
0x2C	EC přehřátí termistoru motoru
0x2D	ED přehřátí termistoru oleje
0x18	E24V výpadek napájení 24V
0x19	E48 výpadek napájení 48V
0x34	E14 neotevření šachetních dveří do časového limitu

Tabulka A.1: Význam chybových kódů

Význam konfiguračních parametrů registrů B0 - BF

Uvedené registry jsou 4-bitové. Hodnota 0000 znamená nastavení registru na 0. Hodnota 0001 udává nastavení registru na 1.

Číslo registru	Význam registru	Význam nastavené hodnoty při nastavení registru na 1	Význam nastavené hodnoty při hodnotě registru 0
0xA0	B0L řízení výahu	simplex, 2 tlačítka	sběr směrem dolů, 1 tlačítko
0xB0	B0H volby	paměť voleb ANO	paměť voleb NE
0xA1	B1L parkování	parkovat	neparkovat
0xB1	B1H spomalení	pouze u MB1	od MB1 a MB2
0xA2	B2L MB1, MB2	rozpínací	spínací
0xB2	B2H, SKRD, SKRH	rozpínací	spínací
0xA3	B3L, DPK, DPKZ	rozpínací	spínací
0xB3	B3H, DP (MB2H)	rozpínací	spínací
0xA4	B4L revizní jízda	vysoká rychlost	nízká rychlost
0xB4	B4H revizní jízda	po SKRD(H) a MB1	po SKRD(H)
0xA5	B5L počítání displeje	bez 0	včetně 0
0xB5	B5H počítání displeje	po dvou	po jednom
0xA6	B6L polohovk	protokol CLK+DAT	protokol RXD
0xB6	B6H LCD/LED	řízení Mx-LCD	řízení FLX-LED
0xA7	B7L termostat	rozpínací	spínací
0xB7	B7H automatické dveře	otevřené ve stanici	zavřené ve stanici
0xA8	B8L kabina a šachta	dveře auto./auto.	dveře auto./mechan.
0xB8	B8H povel u dveří	při jízdě držet	při jízdě nedržet
0xA9	B9L simulace vážení	DPKZ ANO	DPKZ NE
0xB9	B9H simulace	počet přistavení 1x	počet přistavení 2x
0xAA	BAL duplex slave	dva přivolávače	jeden přivolávač
0xBA	BAH duplex master	přednost základního	symetrický
0xAB	BBL duplex master	přízemí zapnutý	přízemí vypnutý
0xBB	BBH světlo stykače	řízení pozitivní	řízení negativní
0xAC	BCL protokol	NBCD ZEVA	BINAR VEGA
0xBC	BCH pohon výtahu	dvou-rychlostní	jedno-rychlostní
0xAD	BDL kabinové dveře	automatické	mechanické
0xBD	BDH pohyblivá	pohyblivá	pevná
0xAE	BEL MUX voleb	zapnutý	vypnutý
0xBE	BEH rízení frekv. měniče	lineátní	binární

Tabulka A.2: Význam konfiguračních parametrů registrů B0-Bf

Význam konfiguračních parametrů registrů D0 - DB

Uvedené registry jsou 8-bitové. Jejich možné hodnoty jsou uvedeny ve sloupci hodnota registru.

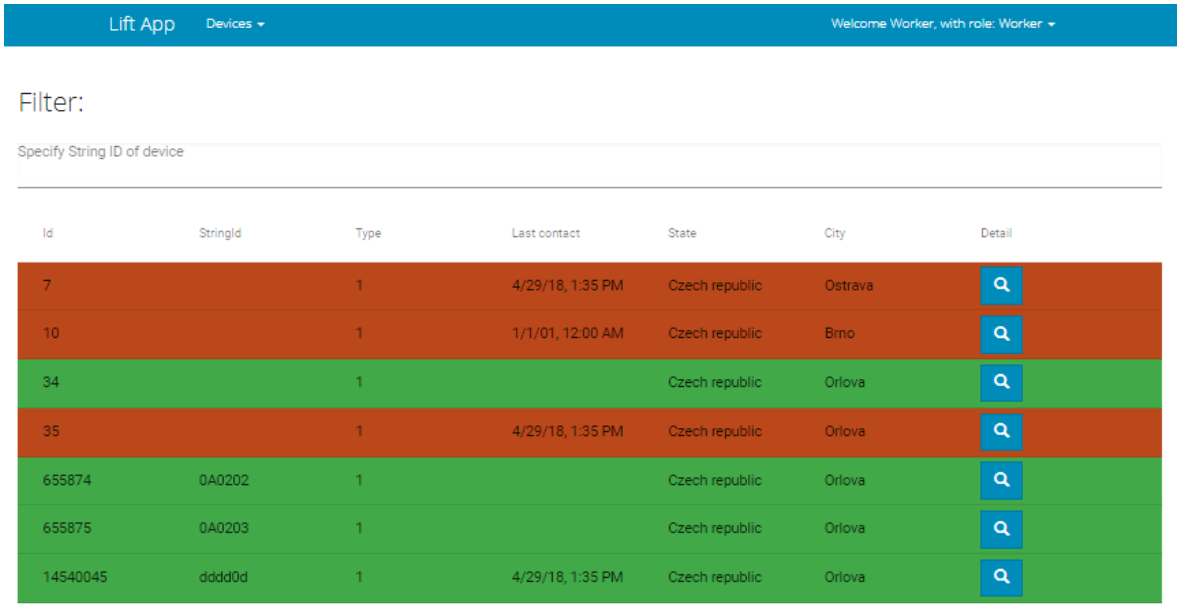
Číslo registru	Význam registru	Hodnota registru	Vysvětlení zobrazení
0xD0	D0 počet stanic výtahu	2 - 16	2 - 16 stanic
0xD1	D1 1. krátká stanice	2 - 16	2. - 16. stanice
0xD2	D2 parkovací stanice od spodu	0 - 7	0. - 7. stanice
0xD3	D3 parkování po době	1 - 16	(1 - 16) * 0.5 minuty
0xD4	D4 počet suterénů	0 - 3	0 - 3 suterény
0xD5	D5 doba otevřených dveří	(1 - 10) * 2	(1 - 10) * 2 sekundy
0xD6	D6 doba startu hvězda/trojůhelník	3 - 25	(3 - 25) * 0.1 minuty
0xD7	D7 evakuační stanice	0 - 7	0 - 7 stanice
0xD8	D8 max. doba jízdy vysokou rychlostí	(1 - 13) * 5	(1 - 13) * 5 sekund
0xD9	D9 max. doba jízdy nízkou rychlostí	(1 - 13) * 5	(1 - 13) * 5 sekund
0xDA	DA čas svícení světla v kabine	1 - 16	(1 - 16) * 30 sekund
0xDB	DB zavření dveří od fotozávory	4, 6, 8	4, 6, 8 sekund

Tabulka A.3: Význam konfiguračních parametrů registrů D0-DB

Příloha B

Podoba systému

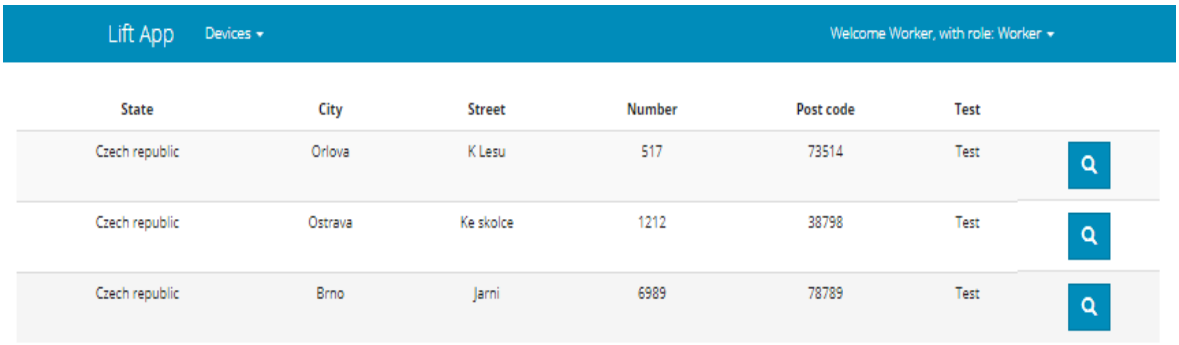
Zobrazení všech zařízení



Obrázek B.1: Zobrazení všech zařízení

Poslední řádek uvádí záznam řídicí jednotky s identifikátorem dddd0d. Ostatní data byla přidána do systému pouze pro přehled. Po uvedení identifikátoru lze zařízení filtrovat. Červeně označené řádky označují zařízení v chybovém stavu. Řádky, které jsou označené zeleně určují, že je zařízení v pořádku. Po kliknutí na symbol lupy se zobrazí detail zařízení.

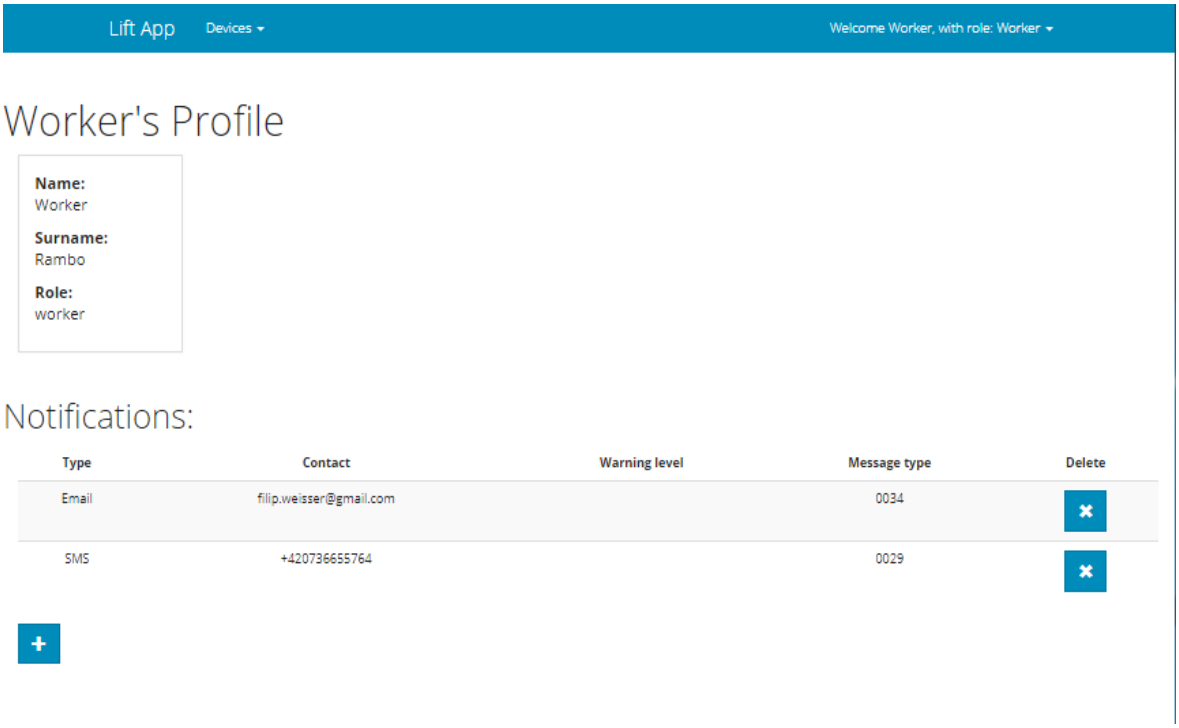
Zobrazení lokací zařízení



Obrázek B.2: Zobrazení lokací zařízení

Obrázek uvádí příklad výpisu lokací firmy. Po kliknutí na tlačítko s logem lupy jsou zákazníkovi zobrazeny všechny zařízení v dané lokalitě. Podoba stránky se zařízeními je zobrazena ve stejné podobě, jaká je ukázána na obrázku B.1. Velmi podobným způsobem vypadá přehled všech typů zařízení dané firmy.

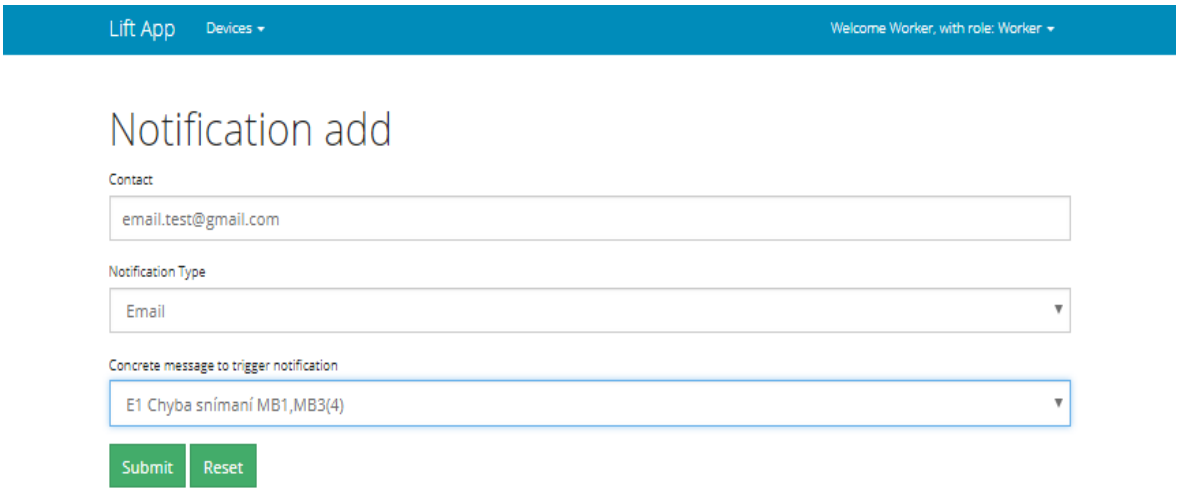
Přehled profilu technika



Obrázek B.3: Přehled profilu technika

V přehledu profilu lze vidět základní informace o technikovi. Technik může smazat vytvořenou notifikaci nebo si přidat novou notifikaci kliknutím na tlačítko se symbolem "+".

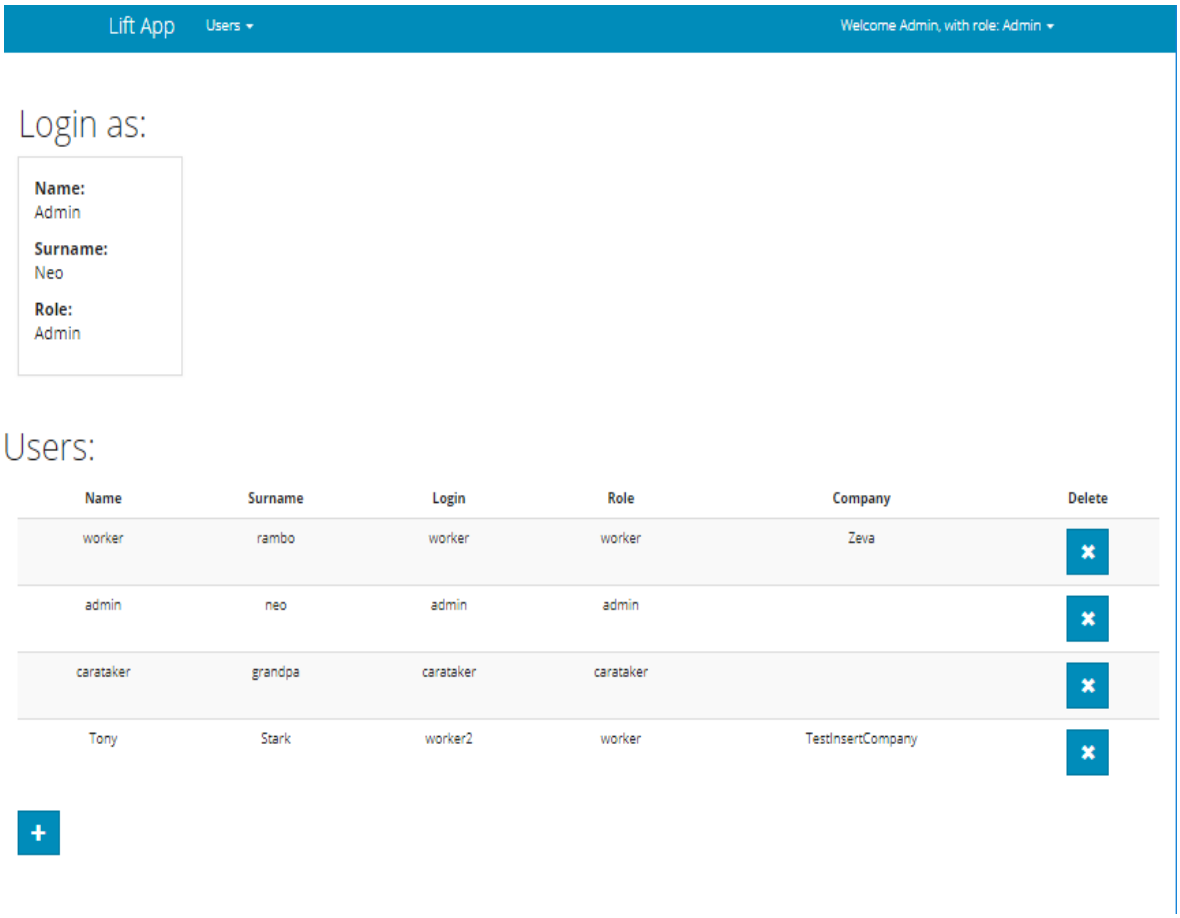
Přidání notifikace technika



Obrázek B.4: Přidání notifikace

Na obrázku je zvolena možnost spuštění notifikace při konkrétní chybě. Možnost spuštění notifikace při všech chybách není zobrazena, neboť současné nastavení obou voleb není povoleno. Při vybrání konkrétní chyby je pro zobrazení druhé možnosti nastavení nutné zmáčknout tlačítko reset. Při zvolení zasílání notifikací při všech chybách je uživateli naopak schována možnost nastavení zasílání notifikací při konkrétní chybě.

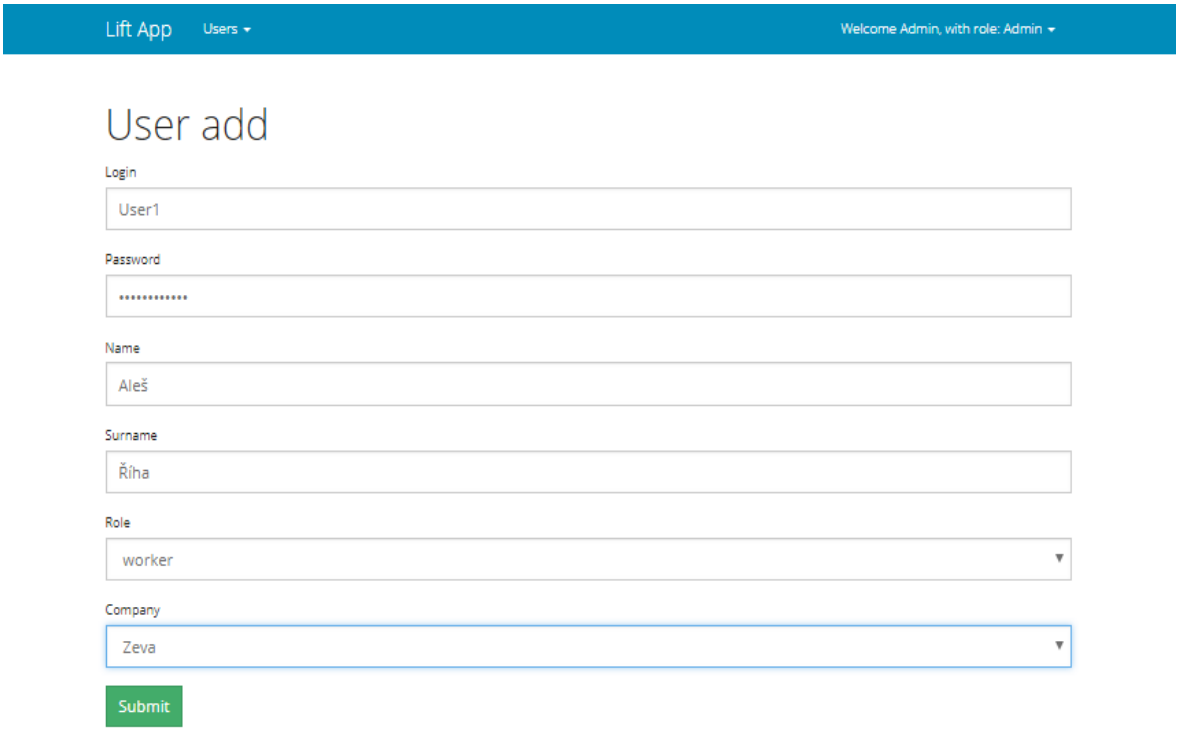
Přehled uživatelů



Obrázek B.5: Zobrazení lokací zařízení

Na obrázku lze vidět přehled všech uživatelů v systému. Možnost zobrazení této stránky má pouze administrátor. Administrátor může smazat uživatele ze systému a odebrat mu tak možnost přihlášení nebo vytvořit nového uživatele zvolením tlačítka "+".

Přidání uživatelů do systému



Obrázek B.6: Zobrazení lokací zařízení

Na obrázku lze vidět příklad přidání uživatele s rolí technika. Technikovi je přiřazena firma Zeva. Možnost přiřazení firmy je zobrazena až po zvolení role pracovníka. Domovníkovi na této stránce nelze přiřadit zařízení a je nutné mu zařízení přidat v databázi. Rozšířené možnosti, které bude mít administrátor k dispozici jsou uvedeny v závěru diplomové práce.

Příloha C

Obsah CD

/src/	adresář s projektem
/doc/	písemná práce ve formátu pdf a její zdrojový dokument
/scripts/	skripty potřebné pro zprovoznění programu práce
README.txt	soubor s návodem pro instalaci a spuštění